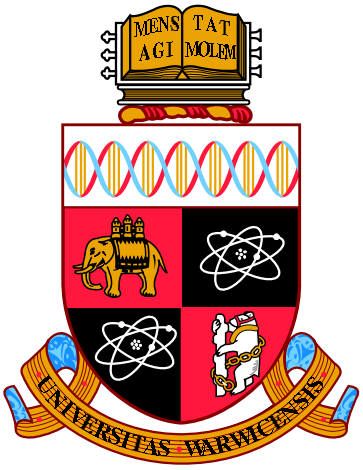

Modular symbols over number fields

by

Maria Teresa Aranés

Thesis

Submitted to the University of Warwick

for the degree of

Doctor of Philosophy

Department of Mathematics

December 2010

THE UNIVERSITY OF WARWICK

Contents

Acknowledgments

First, I would like to thank my supervisor Prof. John E. Cremona, for suggesting this topic and for all his help and advice over the course of my PhD.

I would also like to thank Prof. Angela Arenas, for her encouragement and support back in Barcelona and through all these years.

Finally, I would like to thank my friends and family. Among many others, thanks to Adam, Carlos, Anna Morra, Homero and Marta, Nook, Jorge and Eleonora for their support during my years in Warwick.

This thesis was supported by the European Community under the Marie Curie Training Network GTEM (MRTN-CT-2006-035495).

Declarations

I declare that, to the best of my knowledge and unless otherwise stated, all the work in this thesis is original. The material in this thesis is submitted to the University of Warwick for the degree of Doctor of Philosophy, and has not been submitted to any other university or for any other degree.

Abstract

Let K be a number field, R its ring of integers. For some classes of fields, spaces of cusp forms of weight 2 for $\mathrm{GL}(2, K)$ have been computed using methods based on modular symbols. J.E. Cremona [9] began the programme of extending the classical methods over $\mathbb{Q}$ to the case of imaginary quadratic fields. This work was continued by some of his Ph.D. students [35, 6, 22], and results have been obtained for some imaginary quadratic fields with small class number. More recently, P. Gunnells and D. Yasaki [18] have developed related algorithms for real quadratic fields.

The aim of this thesis is to contribute to the extension of the modular symbols method, when possible developing algorithms and implementations for effective computations. Some parts of the theory are purely algebraic and can be extended to all number fields. We generalise the theory for cusps and Manin symbols; we also describe a generalisation of Atkin-Lehner involutions and study other normaliser elements. On the other hand, all previous explicit computations for the imaginary quadratic field case were done only for specific fields. In the last part of this thesis we begin work towards a general implementation of the techniques used in this case. In particular, we are able to compute a fundamental domain of the hyperbolic 3-space for any imaginary quadratic field.

Implementations of the algorithms described in this thesis have been written by the author in the open-source mathematics software *Sage* [31].

Introduction

Let K be a number field, R its ring of integers. For some classes of fields, spaces of cusp forms of weight 2 for $\mathrm{GL}(2,K)$ have been computed using methods based on modular symbols. That is the case for some imaginary quadratic fields with small class number [10, 35, 6, 22] in which the classical methods over $\mathbb{Q}$ have been extended. Also, algorithms based on modular symbols have been developed for real quadratic fields [18]. For the latter cases, there is also a different method by L. Dembélé [14, 15] which uses some of the techniques of the modular symbols approach.

The aim of this thesis is to contribute to the extension of the modular symbols method. Some parts of the theory which are purely algebraic can be extended to apply to all number fields. On the other hand, some of the techniques used in the imaginary quadratic case have been only applied to computations in particular cases, never having been implemented in a systematic way.

The modular symbols method is based on the duality between homology and the space of cusp forms. Let us sketch the method when K is an imaginary quadratic field. Take $\mathrm{GL}(2,K)$, $\Gamma = \mathrm{GL}(2,R)$ the corresponding multiplicative groups of $\mathrm{Mat}_2(K)$ and $\mathrm{Mat}_2(R)$, the algebras of 2×2 matrices with entries in K or R. We denote by R^* the set of nonzero elements of R, and by $R^\times$ the unit group. For a nonzero ideal $\mathfrak{n}$ of R, that we call *level*, we define the congruence subgroup

$$\Gamma_0(\mathfrak{n}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma \,|\, c \in \mathfrak{n} \right\}.$$

Let $S_2(\mathfrak{n})$ be the space of cusp forms of weight 2 for $\Gamma_0(\mathfrak{n})$. Define the hyperbolic 3-space $\mathcal{H}_3 = \mathbb{C}\times\mathbb{R}_+$ and let $X_0(\mathfrak{n}) = \Gamma_0(\mathfrak{n})\backslash\mathcal{H}_3$, which is a 3-manifold if $\Gamma_0(\mathfrak{n})$ has no trivial elements of finite order.

When the class number of the field is trivial, $h_K = 1$, the 1-homology $H_1(X_0(\mathfrak{n}), \mathbb{R})$ is dual to the space $S_2(\mathfrak{n})$. What is more, this duality is not just of real vector spaces but of modules for the Hecke algebra. Hence we can obtain information about the eigenforms and eigenvalues on $S_2(\mathfrak{n})$ through the Hecke action on the homology. In particular, instead of computing the space of cusp forms, one computes the 1-homology $H_1(X_0(\mathfrak{n}), \mathbb{R})$ via modular symbols and then use the fact that both spaces are dual to one another.

When $h_K > 1$, the domain of the modular forms is in general the union of h_K disjoint copies of $\mathfrak{H}_3$. However, it is possible to obtain all the information about the cusp forms at level $\mathfrak{n}$ by looking only at the homology of the so called principal component, which is simply the space $X_0(\mathfrak{n})$. Hence we still only need to compute the homology for $X_0(\mathfrak{n})$.

To compute the 1-homology group we begin by finding a tessellation of the hyperbolic 3-space $\mathcal{H}_3$, which plays the same role as the upper half complex plane in the classical case $K = \mathbb{Q}$. In order to do that we must find the vertices and edges of this tessellation, which is formed by hyperbolic polyhedra. The vertices are given by the cusps, and the edges, which consist of certain geodesic paths between cusps, can be expressed in terms of Manin symbols. The group Γ acts on this tessellation, so that we have different Γ-orbits of cusps, edges and faces of the polyhedra. From these we will read off relations between the modular symbols which will encode all the information necessary to carry out the homology calculations.

The theory for cusps and Manin symbols is purely algebraic and so it can be generalised to apply to all number fields. In the first chapter we give a review of the results in the classical case $K = \mathbb{Q}$ and then proceed to describe their generalisation to the number field case. We have also written explicit algorithms for much of the theory.

In Chapter 2 we study elements which normalise the congruence subgroup $\Gamma_0(\mathfrak{n})$. We review previous results on the normaliser of Γ in $\mathrm{GL}(2, K)$, which can now be expressed in a more concise way by using some of the theory introduced in Chapter 1. Then in §2.3 we present some results on the normaliser of $\Gamma_0(\mathfrak{n})$, after having a careful look at the classical theory over $\mathbb{Q}$.

In Chapter 3 we restrict our study to the imaginary quadratic number field case. Although some of the techniques in [6, 22] can be used in general, explicit computations have been done only for particular cases. This mostly seems to be due to the differences in the geometry for each field. In this chapter we describe a

method to determine a fundamental region for $\mathfrak{H}_3$ under the action of the group Γ (due to Swan [32]) and our own implementation of the method. This fundamental region provides the ingredients to construct a tessellation of the 3-hyperbolic space by hyperbolic polyhedra.

Finally, in Appendix A we present our implementations of the algorithms described in Chapters 1 and 3.

Chapter 1

Cusps and Manin symbols

In this chapter we generalise the classical theory for cusps and Manin symbols. Due to its algebraic nature, the theory can be applied to all number fields. In §1.1 we give an overview of the classical case (over $\mathbb{Q}$), which serves as a guideline on the strategies used in the general number field case. A detailed exposition of most of the material in section 1.1 can be found in [13] and [30].

All the algorithms and tests described in §1.2 have been implemented by the author and the code is included in the official *Sage* release [31]. Examples of this implementation have been added throughout the section, but a more complete description can be found in the corresponding entries of *Sage's Reference Manual* (see Appendix A.1).

1.1 Cusps and Manin symbols over $\mathbb{Q}$

By a *cusp* of $\mathbb{Q}$ we mean an element of $\mathbb{P}^1(\mathbb{Q}) = \mathbb{Q} \cup \{\infty\}$. Each cusp $\alpha \in \mathbb{P}^1(\mathbb{Q})$ may be represented as $\alpha = p/q$, with $\gcd(p, q) = 1$ and this representation is unique up to multiplication of p and q by -1.

Let $\Gamma = \mathrm{SL}(2, \mathbb{Z})$ and for a positive integer N define the congruence subgroup $\Gamma_0(N)$ as usual:

$$\Gamma_0(N) \;=\; \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma : \; c \equiv 0 \,(\mathrm{mod}\ N) \right\} .$$

Both Γ and $\Gamma_0(N)$ act on the set of cusps by linear fractional transformations:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \left(\frac{p}{q}\right) = \frac{ap + bq}{cp + dq}, \quad \text{for } \gamma = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma.$$

Remark 1.1.1. Linear fractional transformations preserve the coprime condition between numerator and denominator.

It is not difficult to see that the action of Γ is transitive on the set of cusps. We give an explicit proof below, since the main idea used is the key for our strategy in looking at $\Gamma_0(N)$-action.

Proposition 1.1.2. *All cusps over $\mathbb{Q}$ are Γ-equivalent.*

Proof. Let α be a cusp of $\mathbb{Q}$ with representative p/q. There exist $r, s \in \mathbb{Z}$ such that $ps - qr = 1$. We can then complete the column vector $\binom{p}{q}$ to a matrix

$$M_\alpha = \begin{pmatrix} p & r \\ q & s \end{pmatrix} \in \Gamma,$$

which satisfies $M_\alpha \cdot \infty = p/q$. The matrix M_α is unique up to right multiplication by a power of the translation matrix,

$$T = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}. \tag{1.1.1}$$

In particular, given any cusps $\alpha_1, \alpha_2 \in \mathbb{P}^1(\mathbb{Q})$ we have that $\left(M_{\alpha_2} M_{\alpha_1}^{-1}\right) \alpha_1 = \alpha_2$. □

Now we see that given a cusp p/q, we may regard the corresponding column vector $\binom{p}{q}$ as the first column of a matrix in Γ. Hence we will study the action of $\Gamma_0(N)$ on $\mathbb{P}^1(\mathbb{Q})$ via its action by left multiplication on Γ itself. The next result is used to determine right coset representatives for $\Gamma_0(N)$ in Γ.

Proposition 1.1.3. *For $j \in \{1, 2\}$ let $M_j = \begin{pmatrix} a_j & b_j \\ c_j & d_j \end{pmatrix} \in \Gamma$. The following are equivalent:*

(i) *the right cosets $\Gamma_0(N)M_1$ and $\Gamma_0(N)M_2$ are equal,*

(ii) $c_1 d_2 \equiv c_2 d_1 \pmod{N}$,

(iii) $c_1 \equiv u c_2$ *and* $d_1 \equiv u d_2 \pmod{N}$, *with* $\gcd(u, N) = 1$.

Proof. [13, Proposition 2.2.1]. □

We can use Proposition 1.1.3 to deduce a similar result concerning representatives of $\Gamma_0(N)$-equivalence classes of cusps, and we will discuss this in detail in the next

section. Now we proceed instead to introduce Manin symbols.

Observe that the last two conditions given in the Proposition above involve only the bottom row of the matrices in Γ. This leads to the following definition:

Definition 1.1.4. An *M-symbol* or *Manin symbol of level* N is a pair $(c : d)$ such that $\gcd(c, d, N) = 1$, which represents the equivalence class of $(c, d) \in \mathbb{Z}^2$ modulo the relation:

$$(c_1, d_1) \sim (c_2, d_2) \Longleftrightarrow c_1 d_2 \equiv c_2 d_1 \,(\mathrm{mod}\, N).$$

The set of the M-symbols modulo N is $\mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$, the projective line over the ring of integers modulo N. Given an M-symbol $(c : d)$, the integers c and d are only determined modulo N, and we can always choose them such that $\gcd(c, d) = 1$.

Now looking again at Proposition 1.1.3 we see that there is a bijection between our newly defined Manin symbols of level N and right coset representatives for $\Gamma_0(N)$ in Γ:

$$\begin{array}{rcl} \mathbb{P}^1(\mathbb{Z}/N\mathbb{Z}) & \longleftrightarrow & [\Gamma : \Gamma_0(N)] \\ (c : d) & \leftrightarrow & M = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \end{array}$$

where $a, b \in \mathbb{Z}$ are such that $ad - bc = 1$. Note that a different choice of a, b has the effect of multiplying M by a power of the translation matrix T (1.1.1); this does not change the right coset of M, since $T \in \Gamma_0(N)$ for all N.

Remark 1.1.5. We have seen that we may obtain a set of representatives for $\Gamma_0(N)$ in Γ by lifting each M-symbol arbitrarily to an element of Γ. This is a key idea in the computations of homology spaces via modular symbols (see for instance [13, §2.3 and §2.4] or [30, Chapter 3]). Note as well that the natural right coset action of Γ on $[\Gamma : \Gamma_0(N)]$ induces an action on $\mathbb{P}^1(\mathbb{Z}/N\mathbb{Z})$:

$$(c : d) \begin{pmatrix} p & q \\ r & s \end{pmatrix} = (cp + dr : cq + ds).$$

This formula is also very important in modular symbol calculations.

1.1.1 Cusp equivalence under $\Gamma_0(N)$

As mentioned above, from Proposition 1.1.3 we obtain the following corollary which determines $\Gamma_0(N)$-equivalence between cusps:

Proposition 1.1.6. *Let α_1 and α_2 be cusps with representatives p_1/q_1 and p_2/q_2. The following are equivalent:*

(i) $\alpha_2 = M(\alpha_1)$ *for some* $M \in \Gamma_0(N)$.

(ii) $q_2 \equiv uq_1 \pmod{N}$ *and* $up_2 \equiv p_1 \pmod{\gcd(q_1, N)}$, *with* $\gcd(u, N) = 1$.

(iii) $s_1 q_2 \equiv s_2 q_1 \pmod{\gcd(q_1 q_2, N)}$, *where* s_j *satisfies* $p_j s_j \equiv 1 \pmod{q_j}$.

Proof. See [13, Proposition 2.2.3]. □

Remark 1.1.7. Condition (iii) in Proposition 1.1.6 is an efficient test to check if two given rational cusps are $\Gamma_0(N)$-equivalent, and it is the one used in practice. Furthermore, from the proof of the Proposition 1.1.6 it is not difficult to write an algorithm that explicitly computes the transformation matrix between two cusps (when they are $\Gamma_0(N)$-equivalent), as done in [30, Algorithm 8.14].

Using Proposition 1.1.6 it is possible to determine the number of $\Gamma_0(N)$-equivalence classes. We need only to observe that:

Corollary 1.1.8. *Let p_1/q_1 and p_2/q_2 be cusps in the same $\Gamma_0(N)$-orbit. Then $\gcd(q_1, N) = \gcd(q_2, N)$. If $q_1 = q_2 = q$, with $q \nmid N$, $d = \gcd(q, N)$, the following are equivalent:*

- *The cusps p_1/q and p_2/q are $\Gamma_0(N)$-equivalent,*
- $p_1 \equiv p_2 \pmod{\gcd(d, N/d)}$.

Proof. It is clear that $\gcd(q_1, N) = \gcd(q_2, N)$ since, as stated in part (ii) of the previous result, q_1 and q_2 satisfy that $q_2 \equiv uq_1 \pmod{N}$ with $\gcd(u, N) = 1$. For the second part, note that p_1/q and p_2/q are $\Gamma_0(N)$-equivalent if and only if $qs_1 \equiv qs_2 \pmod{\gcd(q^2, N)}$, where $s_i, r_i \in \mathbb{Z}$ and $p_i s_i - q r_i = 1$ for $i = 1, 2$. Now with $d = \gcd(q, N)$ we have

$$\begin{aligned} q(s_1 - s_2) \equiv 0 \pmod{\gcd(q^2, N)} &\Leftrightarrow s_1 - s_2 \equiv 0 \pmod{\gcd(d, N/d)} \\ &\Leftrightarrow p_1 - p_2 \equiv 0 \pmod{\gcd(d, N/d)}. \end{aligned}$$

□

From the statements in Corollary 1.1.8 it is not difficult to write a formula for the number of $\Gamma_0(N)$-equivalence classes and an algorithm to enumerate a list of representatives. However, this direct approach is not so straightforward in the general number field case, where we will use Manin symbols instead. We now

describe the M-symbols approach for the rational case (the same ideas were used by Shimura in [29, Proposition 1.43]).

Counting $\Gamma_0(N)$-equivalence classes

Observe first that our only Γ-orbit of cusps splits into a finite union of $\Gamma_0(N)$-sub-orbits, which are in bijection with the set of double cosets $\Gamma_0(N)\backslash\Gamma/\Gamma_\infty$, where Γ_∞ is the stabiliser of the cusp ∞. Since there is a bijection between M-symbols and $[\Gamma : \Gamma_0(N)]$, we only need to consider the right action of Γ_∞ on M-symbols (in [29, Proposition 1.43], Shimura uses the stabiliser of the cusp 0 instead of Γ_∞). That is, we can enumerate and count $\Gamma_0(N)$-equivalence classes of cusps by enumerating and counting Γ_∞-equivalence classes of M-symbols.

Hence, we need to classify our M-symbols and then give an explicit description of the Γ_∞-action on them. First note that from the definition of Manin symbols one can easily obtain the following properties:

(1) To each class of Manin symbol $(c : d)$ we can associate a divisor of N, namely $\gcd(c, N)$, since:

$$(c : d) = (c' : d') \Longrightarrow \gcd(c, N) = \gcd(c', N).$$

(2) For $c|N$:

$$(c : d) = (c : d') \Longleftrightarrow d \equiv d' \pmod{N/c}.$$

That is, for every c which divides N we have as many different M-symbols of the form $(c : d)$ as non-congruent values of d modulo N/c. Observe as well that for every $(c : d)$ there is an equivalent M-symbol $(c' : d')$ with $c'|N$.

By definition (Remark 1.1.5), the right action of Γ_∞ on M-symbols is given by:

$$(c : d)\begin{pmatrix} 1 & m \\ 0 & 1 \end{pmatrix} = (c : cm + d), \text{ for all } m \in \mathbb{Z}.$$

Clearly, two different M-symbols $(c : d)$, $(c : d')$ are Γ_∞-equivalent if and only if $c|(d - d')$. In particular: for every $(c : d)$, another M-symbol $(c : d')$ is in the same Γ_∞-equivalence class for all $d \equiv d' \pmod{\gcd(c, N/c)}$.

We can now write the following algorithm. It returns a set of coset representatives for the $\Gamma_0(N)$-equivalence classes of cusps, obtained by finding a set of representatives

for the Γ_∞-equivalent classes of M-symbols and then lifting each M-symbol to the corresponding cusp.

Algorithm 1.1.9 (Finding coset representatives for $\Gamma_0(N)$-equivalence classes of cusps.)**.** *Given a positive integer N, this algorithm computes a list of representatives for $\Gamma_0(N)$-equivalence classes of cusps.*

Loop over $c|N$:

1. Set $g = \gcd(c, N/c)$.
2. Loop over $d \pmod g$, with $\gcd(d, g) = 1$:
   (a) Lift d to d' such that $\gcd(c, d') = 1$ and $d' \equiv d \pmod g$.
   (b) Find $a, b \in \mathbb{Z}$ such that $ac - bd' = 1$.
   (c) Output a/c.

We might simplify the algorithm slightly. Note that the coefficients d' in the algorithm describe a set of representatives for $(\mathbb{Z}/g\mathbb{Z})^\times$ which are coprime to c. This is in fact the same set described by the coefficients a which we find in step (b). In particular, we could replace steps (b) and (c) above by a single step (b)$'$:

(b)$'$ Output d'/c.

From all our previous observations, it is easy as well to deduce a formula for the number of $\Gamma_0(N)$-equivalence classes of cusps.

Proposition 1.1.10. *Let N be a positive integer. Then the number of $\Gamma_0(N)$-equivalence classes of rational cusps is:*

$$\sum_{d|N} \varphi(\gcd(d, N/d)),$$

where $\varphi(n) = \#(\mathbb{Z}/n\mathbb{Z})^\times$.

1.2 Cusps and cusp equivalence over Number Fields

Some of the theory in this section had already been developed by J. E. Cremona before the start of this thesis, building on special cases covered in the work of his students for specific imaginary quadratic fields (see [35], [6] and [22]). Here we rewrite and complete this work, adding as well comments on the the practical implementation of the results.

Throughout this section, K is a number field with ring of integers R and class number h_K. We denote by R^* the set of nonzero elements of R, and by $R^\times$ the unit group. As in the Introduction, we define $\Gamma = \mathrm{GL}(2, R)$ as the multiplicative group of $\mathrm{Mat}_2(R)$, and for a nonzero ideal $\mathfrak{n}$ of R (which we call *level*) we have the congruence subgroup

$$\Gamma_0(\mathfrak{n}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma \,|\, c \in \mathfrak{n} \right\}.$$

1.2.1 Cusps over a number field

Definition 1.2.1. A *cusp* of K is an element of $\mathbb{P}^1(K) = K \cup \{\infty\}$.

For $h_K = 1$ we may represent cusps in the form a/b where $a, b \in R$ are coprime. This representation is unique up to multiplication of a and b by a unit of R, and things will be very similar to the situation over $\mathbb{Q}$: we can regard the column vector $\binom{a}{b}$ as the first column of a matrix in Γ, and study the action of Γ and its subgroups on $\mathbb{P}^1(K)$ via their action by left multiplication on Γ itself. In the general case (when the class group is non-trivial) we will replace this action by left multiplication on a special kind of matrix.

In general, cusps may be represented in the form a/b with $a, b \in R$ not both zero, but this representation is not unique. Instead of normalizing the representation of cusps, we allow arbitrary representatives. To each representation $\alpha = a/b$ we may associate the ideal $\langle a, b\rangle$ and its class $[\langle a, b\rangle]$.

Proposition 1.2.2. *The following statements hold:*

(i) *If $a/b = a'/b' \in \mathbb{P}^1(K)$, then $[\langle a, b\rangle] = [\langle a', b'\rangle]$, but the ideals $\langle a, b\rangle$ and $\langle a', b'\rangle$ need not be equal.*

(ii) *Given any ideal $\mathfrak{a}$ in $[\langle a, b\rangle]$, there is a representative a'/b' of the cusp a/b such that $\mathfrak{a} = \langle a', b'\rangle$. This pair (a', b') is unique up to multiplication by a unit $u \in R^\times$.*

Proof. Let $a/b = a'/b'$; then $ab' = a'b$. If $b = 0$ (respectively $b' = 0$), then $b' = 0$ ($b = 0$), and both classes are trivial. Otherwise, $b'\langle a, b\rangle = \langle a', b'\rangle b$, so there exists $\lambda \in K^*$ such that $\langle a, b\rangle = \lambda\langle a', b'\rangle$, i.e. $[\langle a, b\rangle] = [\langle a', b'\rangle]$.

Now for (ii), take $\mathfrak{a} \subseteq R$ ideal such that $[\mathfrak{a}] = [\langle a, b\rangle]$. Then there exist nonzero $c, d \in R$ such that $d\mathfrak{a} = c\langle a, b\rangle$. In particular taking $a' = ca/d \in \mathfrak{a}$, $b' = cb/d \in \mathfrak{a}$ we have $a'/b' = a/b$ and $\mathfrak{a} = \langle a', b'\rangle = \frac{c}{d}\langle a, b\rangle$. For the second part of the statement,

let (a'', b'') be a different pair of elements such that $\mathfrak{a} = \langle a'', b'' \rangle$ and $a'/b' = a''/b''$. We may define $u = a''/a' = b''/b'$ (omitting the zero value in the cases $a' = a'' = 0$ or $b' = b'' = 0$). Now $u\mathfrak{a} = \mathfrak{a}$ (since $\mathfrak{a} = \langle a', b' \rangle = \langle a'', b'' \rangle$), so $u \in R^\times$. That is, we have found a unit $u \in R^\times$ with $a'' = ua'$, $b'' = ub'$. □

As a consequence of the above Proposition, we have a well defined ideal class for each cusp.

Definition 1.2.3. For each cusp $\alpha \in \mathbb{P}^1(K)$, we define the *class of the cusp* $[\alpha]$ as the ideal class $[\langle a, b \rangle] \in \mathrm{Cl}(K)$, where $\alpha = a/b$. We say that α is a *principal cusp* if its class is the trivial class (in which case its associated ideal is principal).

Note as well that part (ii) of Proposition 1.2.2 says that for every ideal $\mathfrak{a} \in [\alpha]$, there is a representative a/b of the cusp α whose associated ideal satisfies $\langle a, b \rangle = \mathfrak{a}$. In other words, we can choose our cusp representative among all ideals in the class of the cusp. This freedom of choice will be helpful in the sections below. In particular, since every ideal class contains an ideal coprime to a given ideal [7, Corollary 1.2.11], we will always be able to choose a representative whose associated ideal is coprime to the level.

As in the rational case, we consider the action of Γ and $\Gamma_0(\mathfrak{n})$ on the set of cusps $\mathbb{P}^1(K)$ by linear fractional transformations, which we define as usual:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \lambda \\ \mu \end{pmatrix} = \frac{a\lambda + b\mu}{c\lambda + d\mu}, \quad \text{for } \gamma = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma \quad \text{and} \quad \alpha = \frac{\lambda}{\mu} \in \mathbb{P}^1(K).$$

Now observe that we have a Γ-equivariant natural map:

$$\begin{array}{ccc} R^2 \setminus \{0\} & \longrightarrow & \mathbb{P}^1(K) \\ \binom{a}{b} & \longmapsto & a/b. \end{array}$$

Hence we can study the action of Γ and $\Gamma_0(\mathfrak{n})$ on the set of cusps $\mathbb{P}^1(K)$ by looking at their action by left multiplication on the set of representatives $\binom{a}{b} \in R^2 \setminus \{0\}$. In fact, as in the classical case, it will be more convenient to think of these representatives as columns of a special kind of matrices, which are introduced in the next section.

1.2.2 $(\mathfrak{a}, \mathfrak{b})$-matrices

From the theory of finitely-generated projective R-modules (see for instance [7, Chapter 1]) we know that given $\mathfrak{a}, \mathfrak{b}$ nonzero ideals of R in inverse classes, we have an isomorphism of R-modules $\mathfrak{a} \oplus \mathfrak{b} \cong R \oplus R$.

We can represent elements of $R \oplus R$ as row vectors, with 2×2 matrices acting on the right. In particular, any R-module isomorphism $R \oplus R \to \mathfrak{a} \oplus \mathfrak{b}$ is necessarily given by a matrix in $\mathrm{Mat}_2(R)$. An $(\mathfrak{a}, \mathfrak{b})$-matrix will be then any matrix realising the isomorphism $R \oplus R \cong \mathfrak{a} \oplus \mathfrak{b}$.

Definition 1.2.4. An $(\mathfrak{a}, \mathfrak{b})$-*matrix* is any matrix M in $\mathrm{Mat}(2, R)$ such that

$$(R \oplus R)M = \mathfrak{a} \oplus \mathfrak{b}.$$

We can explicitly construct $(\mathfrak{a}, \mathfrak{b})$-matrices:

Proposition 1.2.5. *Let $\mathfrak{a} = \langle a_1, a_2 \rangle, \mathfrak{b}$ be ideals in inverse classes, with $\mathfrak{a}\mathfrak{b} = \langle g \rangle$. Then $g = a_1 b_2 - a_2 b_1$ with $b_1, b_2 \in \mathfrak{b}$, and $M = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}$ satisfies*

$$(R \oplus R)M = \mathfrak{a} \oplus \mathfrak{b}.$$

Proof. Since each row of M lies in $\mathfrak{a} \oplus \mathfrak{b}$, it is clear that $(R \oplus R)M \subseteq \mathfrak{a} \oplus \mathfrak{b}$. Conversely, take $(a_3, b_3) \in \mathfrak{a} \oplus \mathfrak{b}$. Then $(a_3, b_3) = (x, y)M$, where:

$$(x, y) = (a_3, b_3)M^{-1} = g^{-1}(a_3 b_2 - a_2 b_3, a_1 b_3 - a_3 b_1),$$

so $(x, y) \in R \oplus R$ since $a_i b_j \in \mathfrak{a}\mathfrak{b} = \langle g \rangle$. □

Example 1.2.6. It is clear from Proposition 1.2.5 that we can take as the first column of an $(\mathfrak{a}, \mathfrak{b})$-matrix any two elements which generate the ideal $\mathfrak{a}$. In particular, given a cusp $\alpha = a_1/a_2$, let $\mathfrak{a} = \langle a_1, a_2 \rangle$ and we can complete the corresponding the column vector $\left(\begin{smallmatrix} a_1 \\ a_2 \end{smallmatrix}\right)$ to an $(\mathfrak{a}, \mathfrak{b})$-matrix.

As we see in the following example, our function `ABmatrix` returns such a matrix for any given number field cusp. Note that in the output a 2×2 matrix

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

is displayed as a list of its coefficients $[a, b, c, d]$.

```
sage: k.<a> = NumberField(x^3 + 11)
sage: alpha = NFCusp(k, oo)
sage: alpha.ABmatrix()
[1, 0, 0, 1]

sage: beta = NFCusp(k, 2, a - 1)
sage: M = beta.ABmatrix(); M
[2, a^2 + a, a - 1, -5]
```

It is clear as well that given a cusp α, any associated $(\mathfrak{a}, \mathfrak{b})$-matrix M_α will send the cusp ∞ to α, that is $M_\alpha(\infty) = \alpha$. We can check our example above with the `apply` function for number field cusps. This function takes as input a matrix given as a list of coefficients and computes its action on the given cusp.

```
sage: NFCusp(k, oo).apply(M) == beta
True
```

Let $X_{\mathfrak{a},\mathfrak{b}}$ denote the set of all $(\mathfrak{a}, \mathfrak{b})$-matrices for fixed ideals $\mathfrak{a}$, $\mathfrak{b}$ in inverse classes. We will now give a description of this set. Observe that when $\mathfrak{a} = \mathfrak{b} = R$, an $(\mathfrak{a}, \mathfrak{b})$-matrix M satisfies $(R \oplus R)M = R \oplus R$, and is just an element of Γ. In fact, Γ can be characterised as the stabiliser in $\mathrm{GL}(2, K)$ of the lattice $R \oplus R$. Indeed, our construction of $(\mathfrak{a}, \mathfrak{b})$-matrices generalises the construction of a unimodular matrix with any prescribed column of coprime elements.

Define

$$\Gamma^{\mathfrak{a},\mathfrak{b}} = \left\{ \begin{pmatrix} x & y \\ z & w \end{pmatrix} \mid x, w \in R,\, y \in \mathfrak{a}^{-1}\mathfrak{b},\, z \in \mathfrak{a}\mathfrak{b}^{-1}, xw - yz \in R^\times \right\}.$$

Remark 1.2.7. $\Gamma^{\mathfrak{a},\mathfrak{b}} = \Gamma$ for $\mathfrak{a} = \mathfrak{b}$, and more generally:

$$\mathfrak{b} | \mathfrak{a} \Rightarrow \Gamma^{\mathfrak{a},\mathfrak{b}} \cap \Gamma = \Gamma_0(\mathfrak{a}\mathfrak{b}^{-1}).$$

The next result generalises the characterisation of Γ as the stabiliser of $R \oplus R$.

Proposition 1.2.8. *Let $\mathfrak{a}, \mathfrak{b}$ be two ideals (not necessarily in inverse ideal classes). Then for $\gamma \in \mathrm{GL}(2, K)$:*

$$(\mathfrak{a} \oplus \mathfrak{b})\gamma = \mathfrak{a} \oplus \mathfrak{b} \iff \gamma \in \Gamma^{\mathfrak{a},\mathfrak{b}}$$

Proof. Suppose that $(\mathfrak{a} \oplus \mathfrak{b})\gamma = \mathfrak{a} \oplus \mathfrak{b}$, with $\gamma = \begin{pmatrix} x & y \\ z & w \end{pmatrix}$. Then $x\mathfrak{a} \subseteq \mathfrak{a}$, $y\mathfrak{a} \subseteq \mathfrak{b}$, $z\mathfrak{b} \subseteq \mathfrak{a}$ and $w\mathfrak{b} \subseteq \mathfrak{b}$, so the entries of γ satisfy $x \in R$, $y \in \mathfrak{a}^{-1}\mathfrak{b}$, $z \in \mathfrak{a}\mathfrak{b}^{-1}$ and $w \in R$,

and in particular $\det\gamma = xw - yz \in R$. Now consider γ^{-1}. From $\mathfrak{a}\oplus\mathfrak{b} = (\mathfrak{a}\oplus\mathfrak{b})\gamma^{-1}$ we deduce that $\det\gamma^{-1} = (\det\gamma)^{-1} \in R$. This proves that $\det\gamma \in R^\times$ and so $\gamma \in \Gamma^{\mathfrak{a},\mathfrak{b}}$. □

We need a few more definitions:

$$\begin{aligned}
\Gamma_\infty^{\mathfrak{a},\mathfrak{b}} &= \left\{ \begin{pmatrix} x & y \\ 0 & w \end{pmatrix} \mid x,\, w \in R,\, y \in \mathfrak{a}^{-1}\mathfrak{b},\, xw \in R^\times \right\}; \\
\Gamma_1^{\mathfrak{a},\mathfrak{b}} &= \left\{ \begin{pmatrix} 1 & y \\ 0 & w \end{pmatrix} \mid y \in \mathfrak{a}^{-1}\mathfrak{b},\, w \in R^\times \right\}; \\
\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}} &= \left\{ \begin{pmatrix} 1 & y \\ 0 & 1 \end{pmatrix} \mid y \in \mathfrak{a}^{-1}\mathfrak{b} \right\}.
\end{aligned}$$

Remark 1.2.9. Note that $\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}} \subset \Gamma_1^{\mathfrak{a},\mathfrak{b}} \subset \Gamma_\infty^{\mathfrak{a},\mathfrak{b}} \subset \Gamma^{\mathfrak{a},\mathfrak{b}}$.

We can now give a description of the set $X_{\mathfrak{a},\mathfrak{b}}$ of $(\mathfrak{a},\mathfrak{b})$-matrices.

Proposition 1.2.10. *Let $M_0 \in X_{\mathfrak{a},\mathfrak{b}}$ be arbitrary. Then:*

$$X_{\mathfrak{a},\mathfrak{b}} = \Gamma M_0 = M_0\Gamma^{\mathfrak{a},\mathfrak{b}},$$

Also, the set of $(\mathfrak{a},\mathfrak{b})$-matrices with same first column as M_0 is $M_0\Gamma_1^{\mathfrak{a},\mathfrak{b}}$, and the set of those with same first column and determinant as M_0 is $M_0\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}}$.

Proof. Recall that $\mathfrak{a}\oplus\mathfrak{b} \cong R\oplus R$, and we can compute an explicit $(\mathfrak{a},\mathfrak{b})$-matrix M that realises the isomorphism: $(R\oplus R)M = \mathfrak{a}\oplus\mathfrak{b}$. In particular, for any $M \in X_{\mathfrak{a},\mathfrak{b}}$ we have:

$$(R\oplus R)M = \mathfrak{a}\oplus\mathfrak{b} = (R\oplus R)M_0 \iff (R\oplus R)MM_0^{-1} = R\oplus R,$$

that is, $M \in X_{\mathfrak{a},\mathfrak{b}}$ if and only if $M \in \Gamma M_0$, which proves the first equality. For the second equality, note that

$$(R\oplus R)M = \mathfrak{a}\oplus\mathfrak{b} = (R\oplus R)M_0 \iff (\mathfrak{a}\oplus\mathfrak{b})M_0^{-1}M = \mathfrak{a}\oplus\mathfrak{b},$$

and from the characterization of $\Gamma^{\mathfrak{a},\mathfrak{b}}$ given in Proposition 1.2.8 it follows that

$$M \in X_{\mathfrak{a},\mathfrak{b}} \iff M_0^{-1}M \in \Gamma^{\mathfrak{a},\mathfrak{b}} \iff M \in M_0\Gamma^{\mathfrak{a},\mathfrak{b}}.$$

The last two statements follow from the definitions of $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$ and $\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}}$ and the identity which we just proved, $X_{\mathfrak{a},\mathfrak{b}} = M_0\Gamma^{\mathfrak{a},\mathfrak{b}}$. □

When we have ideals $\mathfrak{a}$, $\mathfrak{b}$ which are both principal, a diagonal matrix is an obvious choice for an $(\mathfrak{a}, \mathfrak{b})$-matrix representative. However, in general we do not have a canonical choice.

Recall again that an $(\mathfrak{a}, \mathfrak{b})$-matrix exists whenever $\mathfrak{ab}$ is principal, and that in fact we can choose as the lower left entry of the matrix any nonzero element of the ideal $\mathfrak{a}$. This leads to the following definition:

Definition 1.2.11. Let $\mathfrak{n}$ be an integral ideal. An $(\mathfrak{a}, \mathfrak{b})$-*matrix of level* $\mathfrak{n}$ is an $(\mathfrak{a}, \mathfrak{b})$-matrix whose lower left entry lies in $\mathfrak{an}$.

This notion of $(\mathfrak{a}, \mathfrak{b})$-matrix of level $\mathfrak{n}$ will be useful in our discussion of normaliser groups in §2.1. We can characterise these matrices as follows:

Proposition 1.2.12. *M is an $(\mathfrak{a}, \mathfrak{b})$-matrix of level $\mathfrak{n}$ if and only if*

$$(R \oplus R)M = \mathfrak{a} \oplus \mathfrak{b} \qquad \textit{and} \qquad (\mathfrak{n} \oplus R)M = \mathfrak{an} \oplus \mathfrak{b}.$$

Proof. We already know that $(\mathfrak{a}, \mathfrak{b})$-matrices are defined by satisfying the first equality. Clearly if M is of level $\mathfrak{n}$ (as in the definition above) then $(\mathfrak{n} \oplus R)M \subseteq \mathfrak{an} \oplus \mathfrak{b}$. To prove equality we proceed as in Proposition 1.2.5. Given $(a_3, b_3) \in \mathfrak{an} \oplus \mathfrak{b}$, consider $(x, y) = (a_3, b_3)M^{-1}$, and write

$$M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

Then

$$(x, y) = (\det M)^{-1}(a_3 d - c b_3, a b_3 - a_3 b),$$

and it follows that $(x, y) \in \mathfrak{n} \oplus R$. Hence $(a_3, b_3) = (x, y)M \in (\mathfrak{n} \oplus R)M$. □

1.2.3 $\Gamma_0(\mathfrak{n})$-action on $(\mathfrak{a}, \mathfrak{b})$-matrices

Consider a cusp α with representative a_1/a_2. We may regard the column vector $\binom{a_1}{a_2}$ as the first column of an $(\mathfrak{a}, \mathfrak{b})$-matrix (recall Example 1.2.6). Hence we can study the action of Γ and its subgroups on the set of representatives of cusps via its action by left multiplication on $(\mathfrak{a}, \mathfrak{b})$-matrices. The next results consider the set $(\mathfrak{a}, \mathfrak{b})$-matrices under this left $\Gamma_0(\mathfrak{n})$-action, and we will use them later in the study of $\Gamma_0(\mathfrak{n})$-equivalence of cusps. They are in fact a generalisation of Proposition 1.1.3, which is the special case $\mathfrak{a} = \mathfrak{b} = R = \mathbb{Z}$.

Proposition 1.2.13. *Let $\mathfrak{a}, \mathfrak{b}$ be ideals in inverse classes, and let*

$$M_1 = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}, \quad M_2 = \begin{pmatrix} a_1' & b_1' \\ a_2' & b_2' \end{pmatrix}$$

be any two $(\mathfrak{a}, \mathfrak{b})$-matrices. The following statements are equivalent:

(i) *$M_2 = \gamma M_1$ with $\gamma \in \Gamma_0(\mathfrak{n})$,*

(ii) *$a_2' b_2 \equiv a_2 b_2' \pmod{\mathfrak{abn}}$,*

(iii) *There exists $u \in R$ coprime to $\mathfrak{n}$ such that*

  (a) *$ua_2 \equiv a_2' \pmod{\mathfrak{an}}$,*

  (b) *$ub_2 \equiv b_2' \pmod{\mathfrak{bn}}$.*

Proof. Let $g = \det M_1$, so that $\mathfrak{ab} = \langle g \rangle$ and $\det M_2 = u_0 g$ with $u_0 \in R^\times$. Define $\gamma = M_2 M_1^{-1}$, then $\det \gamma = u_0$ and $\gamma \in \Gamma$ by Proposition 1.2.10. Write $\gamma = \begin{pmatrix} v & x \\ y & u \end{pmatrix}$; then

$$y = g^{-1}(a_2' b_2 - a_2 b_2').$$

We can easily prove the equivalence of (i) and (ii): $\gamma \in \Gamma_0(\mathfrak{n})$ if and only if $y \in \mathfrak{n}$, which is true if and only if $a_2' b_2 - a_2 b_2' \in g\mathfrak{n}$, that is, if and only if (ii) holds.

Now assume (i) and (ii). We will see that part (iii) also holds. Note that the bottom diagonal entry of γ satisfies:

$$u = g^{-1}(a_1 b_2' - b_1 a_2') \in R.$$

In particular, we can take this element as the u in (iii). From $uv - xy = \det \gamma = u_0$, we deduce that $uv \equiv u_0 \pmod{\mathfrak{n}}$, so that u is coprime to $\mathfrak{n}$. Now we check that the congruences in (iii) hold with this choice of u. From $M_2 = \gamma M_1$ we have:

$$\begin{aligned} a_2' &= a_1 y + u a_2 \in u a_2 + \mathfrak{an}, \\ b_2' &= b_1 y + u b_2 \in u b_2 + \mathfrak{bn}, \end{aligned}$$

since $y \in \mathfrak{n}$ (recall that $\gamma \in \Gamma_0(\mathfrak{n})$).

Finally, assume that (iii) holds. Then $u a_2 b_2' \equiv a_2' b_2' \equiv u a_2' b_2 \pmod{\mathfrak{abn}}$, and it follows that $u a_2 b_2' g^{-1} \equiv u a_2' b_2 g^{-1} \pmod{\mathfrak{n}}$. Since u is coprime to $\mathfrak{n}$, we can divide by u on both sides. Then multiplying by g again gives (ii). □

Proposition 1.2.14. *If any of the equivalent statements of* Proposition 1.2.13 *holds, then there exist $u \in R$ coprime to $\mathfrak{n}$, $u_0 \in R^\times$ and $\mathfrak{d}$ divisor of $\mathfrak{n}$ such that:*

(a) $\langle a_2 \rangle + \mathfrak{an} = \langle a_2' \rangle + \mathfrak{an} = \mathfrak{ad}$,

(b) $ua_2 \equiv a_2' \pmod{\mathfrak{an}}$,

(c) $u_0 a_1 \equiv u a_1' \pmod{\mathfrak{ad}}$.

Conversely, if the above conditions hold, there exists $\gamma \in \Gamma_0(\mathfrak{n})$ such that

$$\gamma M_1 = M_2' = M_2 \begin{pmatrix} 1 & w \\ 0 & 1 \end{pmatrix} \text{ with } w \in \mathfrak{a}^{-1}\mathfrak{b},$$

so that M_2' is another $(\mathfrak{a}, \mathfrak{b})$-matrix with same first column and determinant as M_2.

Proof. Assume that the three equivalent conditions of Proposition 1.2.13 hold. Let $u \in R$ be coprime to $\mathfrak{n}$, and defined by $u = g^{-1}(a_1 b_2' - b_1 a_2')$, as in the proof of Proposition 1.2.13. Recall that $a_2' = ua_2 + a_1 y \in \langle a_2 \rangle + \mathfrak{an}$, so $\mathfrak{an} + \langle a_2' \rangle \subseteq \mathfrak{an} + \langle a_2 \rangle$. By symmetry we have inclusion in the other direction as well. Hence

$$\mathfrak{an} + \langle a_2' \rangle = \mathfrak{an} + \langle a_2 \rangle = \mathfrak{ad}$$

for some $\mathfrak{d} | \mathfrak{n}$, giving (a). Condition (b) is satisfied trivially and we only need to check (c). With $\gamma = \begin{pmatrix} v & x \\ y & u \end{pmatrix}$ and $\det \gamma = u_0$ as in the proof of Proposition 1.2.13, we observe that:

$$u_0 a_1 - u a_1' = (uv - xy)a_1 - u(va_1 + xa_2) = -x(a_1 y + ua_2) \in \mathfrak{an} + \langle a_2 \rangle = \mathfrak{ad},$$

so that $u_0 a_1 \equiv u a_1' \pmod{\mathfrak{ad}}$, with $u_0 \in R^\times$ and $u \in R$ coprime to $\mathfrak{n}$.

Conversely, suppose that (a), (b) and (c) hold. We will now see that the conditions of (iii) in Proposition 1.2.13 are met if we modify the matrix M_2. Note that:

$$\begin{aligned} (ub_2 - b_2')gu_0 &= ub_2(a_1' b_2' - a_2' b_1') - u_0 b_2'(a_1 b_2 - a_2 b_1) \\ &= b_2 b_2'(ua_1' - u_0 a_1) + u_0 b_1 b_2' a_2 - u b_1' b_2 a_2' \in \mathfrak{b}^2 \mathfrak{ad}, \end{aligned}$$

since $ua_1' - u_0 a_1 \in \mathfrak{ad}$ and $a_2, a_2' \in \langle a_2 \rangle + \mathfrak{an} = \mathfrak{ad}$. Dividing by gu_0 we obtain $ub_2 - b_2' \in \mathfrak{bd}$, and now we observe that:

$$\mathfrak{bd} = \mathfrak{a}^{-1}\mathfrak{b}\mathfrak{ad} = \mathfrak{a}^{-1}\mathfrak{b}(\langle a_2' \rangle + \mathfrak{an}) = \mathfrak{a}^{-1}\mathfrak{b}\langle a_2' \rangle + \mathfrak{bn},$$

so in particular there exists $w \in \mathfrak{a}^{-1}\mathfrak{b}$ such that $ub_2 - b_2' - wa_2' \in \mathfrak{bn}$. That is, we have found $w \in \mathfrak{a}^{-1}\mathfrak{b}$ such that $ub_2 - (wa_2' + b_2') \in \mathfrak{bn}$. Now define $b_2'' = wa_2' + b_2'$. Then the matrices M_1 and M_2', given by

$$M_2' = M_2 \begin{pmatrix} 1 & w \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} a_1' & b_1'' \\ a_2' & b_2'' \end{pmatrix},$$

satisfy the conditions in Proposition 1.2.13(iii), so we know there exists $\gamma \in \Gamma_0(\mathfrak{n})$ such that $\gamma M_1 = M_2'$. □

1.2.4 M-symbols

Recall from §1.1 our introduction of Manin symbols over $\mathbb{Q}$ as the bottom row of matrices in $\Gamma = \mathrm{SL}(2, \mathbb{Z})$. In this section we generalise the notion of M-symbol by characterising which pairs (a, b) occur as the row of an $(\mathfrak{a}, \mathfrak{b})$-matrix.

Proposition 1.2.15. *Let* $\gamma_i = \begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix} \in \Gamma$ *for* $i \in \{1, 2\}$. *Then*

$$\Gamma_0(\mathfrak{n})\gamma_1 = \Gamma_0(\mathfrak{n})\gamma_2 \iff c_1 d_2 \equiv c_2 d_1 \pmod{\mathfrak{n}}.$$

Proof. Apply Proposition 1.2.13 to $\mathfrak{a} = \mathfrak{b} = R$. □

Proposition 1.2.15 is a direct generalisation of Proposition 1.1.3, which we used in the rational case to introduce Manin symbols. Hence we could proceed analogously for the general number field case.

Definition 1.2.16. Consider the set of coprime pairs $(c, d) \in R \oplus R$ modulo the equivalence relation:

$$(c_1, d_1) \sim (c_2, d_2) \iff c_1 d_2 \equiv c_2 d_1 \pmod{\mathfrak{n}}.$$

This is simply $\mathbb{P}^1(R/\mathfrak{n})$. We call its elements *M-symbols or* $(c : d)$*-symbols of level* $\mathfrak{n}$.

Remark 1.2.17. As mentioned in Remark 1.1.5, these symbols have been used in explicit computations with modular symbols when $K = \mathbb{Q}$. They were used as well when K is an imaginary quadratic field (see [10, 6, 22]).

However we can further generalise Definition 1.2.16 to include not only rows of matrices in Γ but also rows of $(\mathfrak{a}, \mathfrak{b})$-matrices. Let $\mathfrak{a}, \mathfrak{b}$ be ideals of R in inverse classes. We now characterize the pairs that can occur as a row of an $(\mathfrak{a}, \mathfrak{b})$-matrix.

Proposition 1.2.18. *Let $\mathfrak{a}, \mathfrak{b}$ be ideals in inverse classes. A pair $(a, b) \in \mathfrak{a} \oplus \mathfrak{b}$ occurs as a row of an $(\mathfrak{a}, \mathfrak{b})$-matrix if and only if*

$$a\mathfrak{a}^{-1} + b\mathfrak{b}^{-1} = R.$$

Proof. Note first that $a\mathfrak{a}^{-1} + b\mathfrak{b}^{-1} = R \Leftrightarrow a\mathfrak{b} + b\mathfrak{a} = \mathfrak{a}\mathfrak{b}$. In particular, taking $\mathfrak{a}\mathfrak{b} = \langle g \rangle$, we have:

$$a\mathfrak{a}^{-1} + b\mathfrak{b}^{-1} = R \Leftrightarrow g \in a\mathfrak{b} + b\mathfrak{a}.$$

Now let M be an $(\mathfrak{a}, \mathfrak{b})$-matrix $\begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}$, with $\det M = g$. Then

$$g = a_1 b_2 - b_1 a_2 \in a_1\mathfrak{b} + b_1\mathfrak{a},$$

and clearly (a_1, b_1) satisfies the condition. Similarly, $g \in a_2\mathfrak{b} + b_2\mathfrak{a}$, so the condition holds as well for the pair (a_2, b_2).
Conversely, given a pair $(a_1, b_1) \in \mathfrak{a} \oplus \mathfrak{b}$ such that $g \in a_1\mathfrak{b} + b_1\mathfrak{a}$ with $\mathfrak{a}\mathfrak{b} = \langle g \rangle$, we can write $g = a_1 b_2 - b_1 a_2$ for some $a_2 \in \mathfrak{a}$, $b_2 \in \mathfrak{b}$. In particular,

$$\begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} \in X_{\mathfrak{a},\mathfrak{b}},$$

so (a_1, b_1) occurs as a row of an $(\mathfrak{a}, \mathfrak{b})$-matrix. □

Definition 1.2.19. An *M-symbol of level $\mathfrak{n}$ and type* $(\mathfrak{a}, \mathfrak{b})$ is an equivalence class of

$$\left\{(a, b) \in \mathfrak{a} \oplus \mathfrak{b} : a\mathfrak{a}^{-1} + b\mathfrak{b}^{-1} = R\right\} / \sim$$

where:

$$\begin{aligned} (a, b) \sim (a', b') &\iff ab' \equiv a'b \pmod{\mathfrak{a}\mathfrak{b}\mathfrak{n}} \\ &\iff \text{there exists } u \in R \text{ coprime to } \mathfrak{n} \text{ such that} \\ &\quad ua \equiv a' \pmod{\mathfrak{a}\mathfrak{n}}, \\ &\quad ub \equiv b' \pmod{\mathfrak{b}\mathfrak{n}}. \end{aligned}$$

An M-symbol of type (R, R) and level $\mathfrak{n}$ will be called *principal M-symbol of level* $\mathfrak{n}$. Note that the M-symbols in Definition 1.2.16 correspond to principal M-symbols in this more general setting.

Remark 1.2.20. In all our applications we will be free to choose the ideals representing each ideal class, that is, we will be free to choose $\mathfrak{a}, \mathfrak{b}$ so that $\mathfrak{a}\mathfrak{b}$ is coprime to $\mathfrak{n}$. In

particular the equivalence relation defining M-symbols will reduce to:

$$\begin{aligned}(a,b)\sim(a',b') &\iff ab'\equiv a'b \pmod{\mathfrak{n}}\\ &\iff \text{there exists } u\in R \text{ coprime to } \mathfrak{n} \text{ such that}\\ &\quad ua\equiv a' \pmod{\mathfrak{n}},\\ &\quad ub\equiv b' \pmod{\mathfrak{n}}.\end{aligned}$$

Directly from Definition 1.2.19 and Propositions 1.2.18 and 1.2.13 we deduce that there is a bijection between the set of M-symbols and the set of orbits of $\Gamma_0(\mathfrak{n})$ acting on $(\mathfrak{a},\mathfrak{b})$-matrices by left multiplication:

$$\{\text{M-symbols of level } \mathfrak{n} \text{ and type } (\mathfrak{a},\mathfrak{b})\} \longleftrightarrow \{\text{Set of orbits of } \Gamma_0(\mathfrak{n})\backslash X_{\mathfrak{a},\mathfrak{b}}\}.$$

Proposition 1.2.21. *For every pair of ideals $\mathfrak{a},\mathfrak{b}$ in inverse classes, the number of M-symbols of level $\mathfrak{n}$ and type $(\mathfrak{a},\mathfrak{b})$ is $\psi(\mathfrak{n})=[\Gamma:\Gamma_0(\mathfrak{n})]$.*

Proof. Let M be any $(\mathfrak{a},\mathfrak{b})$-matrix. Then we know that $X_{\mathfrak{a},\mathfrak{b}}=\Gamma M=\{\gamma M:\gamma\in\Gamma\}$ (recall Proposition 1.2.10), and in particular for any $\gamma_1,\gamma_2\in\Gamma$ we have:

$$\gamma_1 M=\gamma_0\gamma_2 M \iff \gamma_1=\gamma_0\gamma_2, \text{ with } \gamma_0\in\Gamma_0(\mathfrak{n}).$$

Hence the action of $\Gamma_0(\mathfrak{n})$ on $X_{\mathfrak{a},\mathfrak{b}}$ gives the same number of orbits as the action of $\Gamma_0(\mathfrak{n})$ on Γ, which is $\psi(\mathfrak{n})$. □

Below we will use M-symbols to count the number of $\Gamma_0(\mathfrak{n})$-orbits on cusps (§1.2.7). For this it will be useful to be able to normalise our M-symbols in the following way: relaxing the condition that $a\mathfrak{a}^{-1}$, $b\mathfrak{b}^{-1}$ are coprime. The next result generalises [6, Lemmas 24 and 25], in which the normalisation is done for principal M-symbols.

Proposition 1.2.22. *Let $\mathfrak{a},\mathfrak{b}$ be ideals in inverse classes, both coprime to $\mathfrak{n}$. Given $(a,b)\in\mathfrak{a}\oplus\mathfrak{b}$ such that $a\mathfrak{a}^{-1}+b\mathfrak{b}^{-1}+\mathfrak{n}=R$, there exist $(a',b')\in\mathfrak{a}\oplus\mathfrak{b}$ such that*

$$\begin{aligned}&a'\equiv a \pmod{\mathfrak{n}},\\ &b'\equiv b \pmod{\mathfrak{n}},\\ &a'\mathfrak{a}^{-1}+b'\mathfrak{b}^{-1}=R.\end{aligned}$$

Proof. We follow the proof of [6, Lemma 24]. Assume $b\neq 0$ (otherwise interchange the roles of a and b). We only need to take $b'=b$ and $a'=a+c$, where $c\in\mathfrak{a}\mathfrak{n}$ has to be chosen in the following manner.

Take $\mathfrak{q} = \prod \mathfrak{q}_i$, where $\mathfrak{q}_i$ are the prime ideals such that $\mathfrak{q}_i | b\mathfrak{b}^{-1}$ but $\mathfrak{q}_i \nmid a\mathfrak{a}^{-1}$ (there is a finite number of such prime ideals). Then choose an ideal $\mathfrak{r}$ coprime to $b\mathfrak{b}^{-1}$ in the inverse class to $\mathfrak{a}\mathfrak{q}\mathfrak{n}$, so we will have $\mathfrak{a}\mathfrak{n}\mathfrak{q}\mathfrak{r} = \langle c \rangle$, with $c \in \mathfrak{a}\mathfrak{n}$.

Now we will see that our a', b' satisfy the conditions in the Proposition. The congruences hold trivially, so we only need to check that $b\mathfrak{b}^{-1}$ and $a'\mathfrak{a}^{-1}$ are indeed coprime. Let $\mathfrak{p}$ be a prime dividing $b\mathfrak{b}^{-1}$, we will now prove that $\mathfrak{a}\mathfrak{p} \nmid a'$.

If $\mathfrak{p} | a\mathfrak{a}^{-1}$ then $\mathfrak{p} \nmid \mathfrak{q}$ (by construction of $\mathfrak{q}$), $\mathfrak{p} \nmid \mathfrak{n}$ (since by hypothesis $a\mathfrak{a}^{-1} + b\mathfrak{b}^{-1} + \mathfrak{n} = R$) and $\mathfrak{p} \nmid \mathfrak{r}$ (by construction of $\mathfrak{r}$). Then $\mathfrak{p} \nmid c\mathfrak{a}^{-1} = \mathfrak{n}\mathfrak{q}\mathfrak{r}$, so $\mathfrak{a}\mathfrak{p} \nmid c$, and since $\mathfrak{a}\mathfrak{p} | a$, we have $\mathfrak{a}\mathfrak{p} \nmid a' = a + c$.

If $\mathfrak{p} \nmid a\mathfrak{a}^{-1}$, then $\mathfrak{p} | \mathfrak{q}$, so $\mathfrak{p} | c\mathfrak{a}^{-1} = \mathfrak{n}\mathfrak{q}\mathfrak{r}$, and hence $\mathfrak{a}\mathfrak{p} | c$. But $\mathfrak{a}\mathfrak{p} \nmid a$, so again $\mathfrak{a}\mathfrak{p} \nmid a' = a + c$. □

1.2.5 Cusp equivalence under Γ

Let us go back to our study of cusps. With the theory developed for $(\mathfrak{a}, \mathfrak{b})$-matrices it is now easy to prove the following result:

Proposition 1.2.23. *The ideal $\langle a, b \rangle$ associated to $\binom{a}{b}$ is invariant under the action of Γ. Conversely, if $\langle a, b \rangle = \langle a', b' \rangle$, then $\gamma \binom{a}{b} = \binom{a'}{b'}$ for some $\gamma \in \Gamma$.*

Proof. If $\binom{a'}{b'} = \gamma \binom{a}{b}$ for $\gamma \in \Gamma$, then $a', b' \in \langle a, b \rangle$, so $\langle a', b' \rangle \subseteq \langle a, b \rangle$. Since γ is invertible we have the symmetric result $\langle a, b \rangle \subseteq \langle a', b' \rangle$, so equality follows.
For the converse statement, take $\mathfrak{a} = \langle a, b \rangle = \langle a', b' \rangle$. Let $\mathfrak{b}$ be an ideal in the inverse class to $\mathfrak{a}$, with $\mathfrak{a}\mathfrak{b} = \langle g \rangle$. Consider now M_1, M_2 $(\mathfrak{a}, \mathfrak{b})$-matrices with first columns $\binom{a}{b}$ and $\binom{a'}{b'}$ respectively. Then taking $\gamma = M_2 M_1^{-1} \in \Gamma$ we have $\gamma \binom{a}{b} = \binom{a'}{b'}$. □

We can now prove the following classical result (known in the 19th century to Hurwitz, Humbert and Bianchi), which describes the action of Γ on the set of cusps:

Proposition 1.2.24. *There is a bijection:*

$$\begin{array}{ccc} \Gamma \backslash \mathbb{P}^1(K) & \longrightarrow & Cl(K) \\ \alpha & \longmapsto & [\alpha] \end{array}$$

where if a/b is a representative of the cusp α, $[\alpha] = [\langle a, b \rangle]$.

Proof. The map is well-defined by Proposition 1.2.2, and it is obviously surjective. It is injective since given $a/b, a'/b' \in \mathbb{P}^1(K)$ with $[\langle a, b \rangle] = [\langle a', b' \rangle]$, by Proposition 1.2.2 we may assume they have the same ideal, i.e. $\langle a, b \rangle = \langle a', b' \rangle$. Then by the previous result we know that there exists $\gamma \in \Gamma$ such that $\binom{a'}{b'} = \gamma \binom{a}{b}$, that is, a/b and a'/b' are in the same Γ-orbit. □

In particular, we observe that the action of Γ on the set of cusps is transitive (as it happened in the classical case with $K = \mathbb{Q}$, $R = \mathbb{Z}$) if and only if $h_K = 1$. Note as well that now we can describe the principal cusps (Definition 1.2.3) as the orbit of the cusp infinity under the action of Γ.

1.2.6 Cusp equivalence under $\Gamma_0(\mathfrak{n})$.

Fix a nonzero ideal $\mathfrak{n}$. We are going to describe the orbits of the set of cusps under the action of the congruence subgroup $\Gamma_0(\mathfrak{n})$.

Definition 1.2.25. Let $\alpha = a_1/a_2$ be a cusp of K. The *denominator ideal* $\mathfrak{d}(\alpha)$ is the ideal $\langle a_2\rangle/\langle a_1, a_2\rangle$. Note that this definition is in fact independent of the choice of the representation $\alpha = a/b$.

Now to each cusp α we associate the ideal $\mathfrak{d}_{\mathfrak{n}}(\alpha) = \mathfrak{d}(\alpha) + \mathfrak{n}$. This ideal is well-defined and $\Gamma_0(\mathfrak{n})$-invariant (as we will see in Proposition 1.2.26). Hence we have two necessary conditions for two cusps α, α' to be $\Gamma_0(\mathfrak{n})$-equivalent: they must satisfy $[\alpha] = [\alpha']$ and $\mathfrak{d}_{\mathfrak{n}}(\alpha) = \mathfrak{d}_{\mathfrak{n}}(\alpha')$.

Using the previous results concerning $\Gamma_0(\mathfrak{n})$-action on $(\mathfrak{a}, \mathfrak{b})$-matrices (namely Propositions 1.2.13 and 1.2.14) we can complete our description of $\Gamma_0(\mathfrak{n})$-equivalence classes of cusps. The following result is a generalisation of the results over $\mathbb{Q}$ (Proposition 1.1.6).

Proposition 1.2.26. *Let α, α' be two cusps in the same ideal class and choose representatives $\alpha = a_1/a_2$ and $\alpha' = a_1'/a_2'$ with the same ideal $\mathfrak{a} = \langle a_1, a_2\rangle = \langle a_1', a_2'\rangle$. Then the following are equivalent:*

(i) *$\gamma(\alpha) = \alpha'$ for some $\gamma \in \Gamma_0(\mathfrak{n})$,*

(ii) *there exist $u \in R$ coprime to $\mathfrak{n}$, $u_0 \in R^\times$ and a divisor $\mathfrak{d}$ of $\mathfrak{n}$ such that:*

  (a) $\mathfrak{d}_{\mathfrak{n}}(\alpha) = \mathfrak{d}_{\mathfrak{n}}(\alpha') = \mathfrak{d}$,

  (b) $a_2' \equiv u a_2 \pmod{\mathfrak{n}\mathfrak{a}}$,

  (c) $u a_1' \equiv u_0 a_1 \pmod{\mathfrak{d}\mathfrak{a}}$.

When $\mathfrak{a}$ and $\mathfrak{n}$ are coprime, we can replace (ii) *by the simpler:*

(ii)$'$ *there exist $u \in R$ coprime to $\mathfrak{n}$, $u_0 \in R^\times$ and a divisor $\mathfrak{d}$ of $\mathfrak{n}$ such that:*

  (a) $\langle a_2\rangle + \mathfrak{n} = \langle a_2'\rangle + \mathfrak{n} = \mathfrak{d}$,

  (b) $a_2' \equiv u a_2 \pmod{\mathfrak{n}}$,

  (c) $u a_1' \equiv u_0 a_1 \pmod{\mathfrak{d}}$.

Proof. We first observe that the existence of $\gamma \in \Gamma_0(\mathfrak{n})$ with $\gamma(\alpha) = \alpha'$ is equivalent to the existence of $\gamma \in \Gamma_0(\mathfrak{n})$ with $\gamma \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} a_1' \\ a_2' \end{pmatrix}$. This is because if we choose other representatives for the cusps with the same ideal the difference will be a product by a unit (Proposition 1.2.2), and we are free to multiply γ by a unit times the identity matrix.

Suppose that $\gamma(\alpha) = \alpha'$. We may assume that $\gamma \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} a_1' \\ a_2' \end{pmatrix}$, and construct an $(\mathfrak{a}, \mathfrak{b})$-matrix M_1 with first column $\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$. Define $M_2 = \gamma M_1$. Then M_2 is an $(\mathfrak{a}, \mathfrak{b})$-matrix with first column $\begin{pmatrix} a_1' \\ a_2' \end{pmatrix}$. In particular, we are in the conditions of Proposition 1.2.14 and (ii) holds.

For the converse, assume (ii). Again using Proposition 1.2.14, we see that there exists $\gamma \in \Gamma_0(\mathfrak{n})$ such that $\gamma M_1 = M_2'$, where M_2' is another $(\mathfrak{a}, \mathfrak{b})$-matrix with the same first column and determinant as M_2. In particular, that means

$$\gamma \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} a_1' \\ a_2' \end{pmatrix}.$$

For the last part, we begin by observing that if $\mathfrak{a} + \mathfrak{n} = \langle 1 \rangle$ then clearly

$$\mathfrak{d}_{\mathfrak{n}}(\alpha) = \langle a_2 \rangle \mathfrak{a}^{-1} + \mathfrak{n} = \langle a_2 \rangle + \mathfrak{n}.$$

For the same reason, $\mathfrak{d}_{\mathfrak{n}}(\alpha') = \langle a_2' \rangle + \mathfrak{n}$. Also, since $a_2' - u a_2 \in \mathfrak{a}$ we have

$$a_2' - u a_2 \in \mathfrak{n} \Leftrightarrow a_2' - u a_2 \in \mathfrak{a}\mathfrak{n}.$$

Similarly, $u a_1' - u_0 a_1 \in \mathfrak{d} \Leftrightarrow u a_1' - u_0 a_1 \in \mathfrak{d}\mathfrak{a}$. □

More convenient to test equivalence between two cusps is the following criterion, which generalises Proposition 1.1.6(iii).

Corollary 1.2.27. *Let α, α' be two cusps in the same ideal class and choose representatives $\alpha = a_1/a_2$ and $\alpha' = a_1'/a_2'$ with the same ideal $\mathfrak{a} = \langle a_1, a_2 \rangle = \langle a_1', a_2' \rangle$, which is coprime to the level $\mathfrak{n}$. Let $\mathfrak{b}$ be any ideal in the inverse class to $\mathfrak{a}$, and form $(\mathfrak{a}, \mathfrak{b})$-matrices*

$$M_1 = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}, \quad M_2 = \begin{pmatrix} a_1' & b_1' \\ a_2' & b_2' \end{pmatrix}.$$

Then α and α' are $\Gamma_0(\mathfrak{n})$-equivalent if and only if

(i) $\langle a_2 \rangle + \mathfrak{n} = \langle a_2' \rangle + \mathfrak{n} = \mathfrak{d}$,

(ii) *there exists* $u_0 \in R^\times$ *such that* $a_2'b_2 \equiv u_0 a_2 b_2' \pmod{\mathfrak{abd}^2}$.

Proof. Suppose there exists $\gamma \in \Gamma_0(\mathfrak{n})$ such that $\gamma\alpha = \alpha'$. Without loss of generality we may assume that $\gamma \binom{a_1}{a_2} = \binom{a_1'}{a_2'}$. Then $\gamma M_1 = M_2'$, where M_2' is an $(\mathfrak{a}, \mathfrak{b})$-matrix with same first column as M_2. That is:

$$M_2' = \begin{pmatrix} a_1' & b_1'' \\ a_2' & b_2'' \end{pmatrix} = M_2 \begin{pmatrix} 1 & w \\ 0 & u_0 \end{pmatrix},$$

where $w \in \mathfrak{a}^{-1}\mathfrak{b}$ and $u_0 = (\det\gamma)(\det M_1)(\det M_2)^{-1} \in R^\times$. Now since $\gamma M_1 = M_2'$, from Proposition 1.2.13 we know that $a_2'b_2 - a_2b_2'' \in \mathfrak{abn}$. Since $b_2'' = a_2'w + u_0b_2'$, it follows that

$$a_2'b_2 - u_0a_2b_2' \in \mathfrak{a}^{-1}\mathfrak{b}\langle a_2, a_2'\rangle + \mathfrak{abn}.$$

We will now check that in fact $\mathfrak{a}^{-1}\mathfrak{b}\langle a_2, a_2'\rangle + \mathfrak{abn} = \mathfrak{abd}^2$, with $\mathfrak{d} = \langle a_2\rangle + \mathfrak{n}$. By Proposition 1.2.26, $\mathfrak{d} = \langle a_2\rangle + \mathfrak{n} = \langle a_2'\rangle + \mathfrak{n}$. Since we have chosen $\mathfrak{a}$ coprime to $\mathfrak{n}$ we have $\mathfrak{d}^2 = \langle a_2a_2'\rangle + \mathfrak{n}$ and $\mathfrak{a}^{-2}\langle a_2a_2'\rangle + \mathfrak{n} = \langle a_2a_2'\rangle + \mathfrak{n}$. Hence

$$\mathfrak{a}^{-1}\mathfrak{b}\langle a_2, a_2'\rangle + \mathfrak{abn} = \mathfrak{ab}(\mathfrak{a}^{-2}\langle a_2a_2'\rangle + \mathfrak{n}) = \mathfrak{ab}(\langle a_2a_2'\rangle + \mathfrak{n}) = \mathfrak{abd}^2.$$

For the converse, suppose that (i) and (ii) hold. Since $\mathfrak{d} = \langle a_2\rangle + \mathfrak{n} = \langle a_2'\rangle + \mathfrak{n}$, we again have $\mathfrak{a}^{-1}\mathfrak{b}\langle a_2, a_2'\rangle + \mathfrak{abn} = \mathfrak{abd}^2$. In particular,

$$a_2'b_2 - u_0a_2b_2' \in \mathfrak{a}^{-1}\mathfrak{b}\langle a_2, a_2'\rangle + \mathfrak{abn}.$$

From this we deduce that there exists an $w \in \mathfrak{a}^{-1}\mathfrak{b}$ such that

$$a_2'b_2 - b_2''a_2 \in \mathfrak{abn},$$

where $b_2'' = wa_2' + u_0b_2'$. We can then apply Proposition 1.2.13 and we know there exists $\gamma \in \Gamma_0(\mathfrak{n})$ such that $M_2' = \gamma M_1$ where, as before,

$$M_2' = \begin{pmatrix} a_1' & b_1'' \\ a_2' & b_2'' \end{pmatrix} = M_2 \begin{pmatrix} 1 & w \\ 0 & u_0 \end{pmatrix}.$$

In particular, $\gamma\alpha = \alpha'$. □

Example 1.2.28. Our *Sage* implementation of a test for $\Gamma_0(\mathfrak{n})$-equivalence of cusps is based on Corollary 1.2.27. Take $K = \mathbb{Q}(\sqrt{-5})$, and $\mathfrak{n} = \langle 3\rangle$. With the `is_Gamma0_equivalent` function we can test for equivalence between any two cusps of $\mathbb{P}^1(K)$.

```
sage: k.<a> = NumberField(x^2 + 5)
sage: N = k.ideal(3)
sage: alpha = NFCusp(k, 3, a + 1)
sage: beta = NFCusp(k, 2, a - 3)
sage: alpha.is_Gamma0_equivalent(beta, N)
True
```

It is not difficult to compute the transformation matrix between two equivalent cusps. Note that this is the matrix $\gamma = M_2' M_1^{-1}$ in the proof of Corollary 1.2.27. The function `is_Gamma0_equivalent` will return the transformation matrix[1] (as a list of coefficients) if we set the value of the optional `Transformation` argument to `True`.

```
sage: t, M = alpha.is_Gamma0_equivalent(beta, N, Transformation=True)
sage: M
[-170/7*a - 6/7, 82/7*a + 421/7, 36*a + 63, 13*a - 120]
```

We are now able to distinguish cusps in different equivalence classes. It remains to see how to count and enumerate the number of different $\Gamma_0(\mathfrak{n})$-orbits.

1.2.7 Number of $\Gamma_0(\mathfrak{n})$-equivalence classes.

In the case $\mathfrak{n} = R$, the number of $\Gamma_0(\mathfrak{n})$-equivalence classes is h_K (Proposition 1.2.24). In general, we first observe that from general theory of group actions, each Γ-orbit splits into a finite union of $\Gamma_0(\mathfrak{n})$-sub-orbits, which are in bijection with the set of double cosets $\Gamma_0(\mathfrak{n})\backslash\Gamma/\Gamma_\alpha$, where α is any cusp in the orbit and Γ_α is its stabilizer. That is, for any Γ-orbit we have a decomposition

$$\Gamma = \coprod \Gamma_0(\mathfrak{n})\gamma_i\Gamma_\alpha, \tag{1.2.1}$$

where $\{\gamma_i\}_i$ is a set of double coset representatives for $\Gamma_0(\mathfrak{n})\backslash\Gamma/\Gamma_\alpha$.

It is preferable to reformulate the problem using $(\mathfrak{a}, \mathfrak{b})$-matrices. This will allow us to avoid dealing with the possibly complicated structure of the stabilizer Γ_α.

[1] At the time this thesis was written a bug in the function had not yet been corrected in the official release of *Sage*. In consequence at least up to version 4.5.3 the transformation matrix returned by *Sage* will differ from the correct one computed in the example.

Using $(\mathfrak{a},\mathfrak{b})$-matrices. Fix an ideal class, and let $\mathfrak{a}$ be an ideal in this class, and $\mathfrak{b}$ an ideal in the inverse class. Then all cusps in the class have representations with associated ideal $\mathfrak{a}$. In particular (recall Example 1.2.6) all cusps in the class are of the form $\alpha = M(\infty)$, where M is an $(\mathfrak{a},\mathfrak{b})$-matrix. Now we will see that the $\Gamma_0(\mathfrak{n})$-sub-orbits of $\Gamma\alpha$ are also in bijection with the double cosets $\Gamma_0(\mathfrak{n})\backslash X_{\mathfrak{a},\mathfrak{b}}/\Gamma_1^{\mathfrak{a},\mathfrak{b}}$.

Fix $M_0 \in X_{\mathfrak{a},\mathfrak{b}}$. By Proposition 1.2.10, $X_{\mathfrak{a},\mathfrak{b}} = \Gamma M_0$; now observe that for $\alpha = M_0(\infty)$, we have $M_0\Gamma_\infty^{\mathfrak{a},\mathfrak{b}}M_0^{-1} = \Gamma_\alpha$. Then from our double coset decomposition above (1.2.1) we get:

$$X_{\mathfrak{a},\mathfrak{b}} = \Gamma M_0 = \coprod \Gamma_0(\mathfrak{n})\gamma_i\Gamma_\alpha M_0 = \coprod \Gamma_0(\mathfrak{n})M_i\Gamma_\infty^{\mathfrak{a},\mathfrak{b}},$$

with $M_i = \gamma_i M_0 \in X_{\mathfrak{a},\mathfrak{b}}$ and $\{\gamma_i\}_i$ our choice for a set of representatives of $\Gamma_0(\mathfrak{n})\backslash\Gamma/\Gamma_\alpha$. That is, we have obtained a decomposition:

$$X_{\mathfrak{a},\mathfrak{b}} = \coprod \Gamma_0(\mathfrak{n})M_i\Gamma_\infty^{\mathfrak{a},\mathfrak{b}},$$

with $M_i = \gamma_i M_0$ running through a set of representatives for $\Gamma_0(\mathfrak{n})\backslash X_{\mathfrak{a},\mathfrak{b}}/\Gamma_\infty^{\mathfrak{a},\mathfrak{b}}$. Now by taking into account that $\Gamma_\infty^{\mathfrak{a},\mathfrak{b}} = R^\times\Gamma_1^{\mathfrak{a},\mathfrak{b}}$ we can further simplify our decomposition of $X_{\mathfrak{a},\mathfrak{b}}$ to:

$$X_{\mathfrak{a},\mathfrak{b}} = \coprod \Gamma_0(\mathfrak{n})M_i\Gamma_1^{\mathfrak{a},\mathfrak{b}}.$$

We can then consider $(\mathfrak{a},\mathfrak{b})$-matrices instead of cusps, with our problem reduced to determine a set of double coset representatives for $\Gamma_0(\mathfrak{n})\backslash X_{\mathfrak{a},\mathfrak{b}}/\Gamma_1^{\mathfrak{a},\mathfrak{b}}$. There are two natural different ways to consider the enumeration of these equivalence classes, which we call "vertical approach" and "horizontal approach":

- "Vertical approach": we consider the left action of $\Gamma_0(\mathfrak{n})$ on $X_{\mathfrak{a},\mathfrak{b}}/\Gamma_1^{\mathfrak{a},\mathfrak{b}}$. In this case we are basically looking at the action of $\Gamma_0(\mathfrak{n})$ on column vectors.
- "Horizontal approach": we consider the right action of the stabilizer $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$ on $\Gamma_0(\mathfrak{n})\backslash X_{\mathfrak{a},\mathfrak{b}}$. Since there is a bijection between $\Gamma_0(\mathfrak{n})\backslash X_{\mathfrak{a},\mathfrak{b}}$ and the set of M-symbols of type $(\mathfrak{a},\mathfrak{b})$ and level $\mathfrak{n}$, we are basically looking at the action of $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$ on row vectors.

Following our work for the classical case $K = \mathbb{Q}$ in §1.1.1, we take the "horizontal approach" to prove the next result.

Proposition 1.2.29. *Each Γ-orbit in $\mathbb{P}^1(K)$ splits into $\sum_{\mathfrak{d}|\mathfrak{n}} \varphi_{\mathrm{u}}(\mathfrak{d}+\mathfrak{n}\mathfrak{d}^{-1})$ disjoint $\Gamma_0(\mathfrak{n})$-orbits, with*

$$\varphi_{\mathrm{u}}(\mathfrak{m}) = \#((R/\mathfrak{m})^\times/U_{\mathfrak{m}}),$$

where $U_{\mathfrak{m}}$ denotes the image of $R^\times$ in $(R/\mathfrak{m})^\times$. Hence the total number of $\Gamma_0(\mathfrak{n})$-orbits of cusps is

$$h_K \sum_{\mathfrak{d}|\mathfrak{n}} \varphi_{\mathrm{u}}(\mathfrak{d}+\mathfrak{n}\mathfrak{d}^{-1}).$$

Proof. Fix an ideal class and let $\mathfrak{a}$ be an ideal in this class, coprime to $\mathfrak{n}$. Now choose an ideal $\mathfrak{b}$ in the inverse class, also coprime to $\mathfrak{n}$.

We will show that the number of orbits of cusps in the class $[\mathfrak{a}]$ (which is in fact independent of the class) is given by:

$$\sum_{\mathfrak{d}|\mathfrak{n}} \varphi_{\mathrm{u}}(\mathfrak{d}+\mathfrak{n}\mathfrak{d}^{-1}).$$

As observed above, the number of orbits in the class $[\mathfrak{a}]$ is the same as the number of right $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$-orbits on the set of M-symbols of type $(\mathfrak{a},\mathfrak{b})$ and level $\mathfrak{n}$. We will then begin by classifying our M-symbols of type $(\mathfrak{a},\mathfrak{b})$, which are of the form $(a:b)$ with $(a,b)\in\mathfrak{a}\oplus\mathfrak{b}$ and $a\mathfrak{a}^{-1}+b\mathfrak{b}^{-1}=R$, by the possible values of a.

To each M-symbol $(a:b)$ we can associate the ideal $\mathfrak{d}=\langle a\rangle+\mathfrak{n}=a\mathfrak{a}^{-1}+\mathfrak{n}$, a divisor of $\mathfrak{n}$. This is a well-defined association, since if $(a:b)=(a':b')$, then there exists $u\in R$ coprime to $\mathfrak{n}$ such that $a\equiv ua' \pmod{\mathfrak{n}}$, which implies $\langle a\rangle+\mathfrak{n}=\langle a'\rangle+\mathfrak{n}$.

Conversely, if we fix $\mathfrak{d}$ a divisor of $\mathfrak{n}$, we can always find $\mathfrak{d}'$ coprime to $\mathfrak{n}$ such that $\mathfrak{a}\mathfrak{d}\mathfrak{d}'$ is principal, and we choose a such that $\mathfrak{a}\mathfrak{d}\mathfrak{d}'=\langle a\rangle$. Then $\langle a\rangle+\mathfrak{n}=\mathfrak{d}$, since $\mathfrak{a}\mathfrak{d}'$ is coprime to $\mathfrak{n}$. Now, for any M-symbol $(a':b')$ with $\langle a'\rangle+\mathfrak{n}=\langle a\rangle+\mathfrak{n}$, there exists $u\in R$ coprime to $\mathfrak{n}$ such that $a\equiv ua' \pmod{\mathfrak{n}}$ and, in particular,

$$(a':b')=(ua':ub')=(a:b) \quad \text{for } b=ub'.$$

Thus we have shown that every M-symbol associated with the fixed divisor $\mathfrak{d}$ has the form $(a:b)$ with the value of a that we have fixed and some $b\in\mathfrak{b}$. Note that in this last step we required the validity of M-symbols $(a:b)$ satisfying the weaker condition of Proposition 1.2.22, namely that $a\mathfrak{a}^{-1}+b\mathfrak{b}^{-1}$ is coprime to $\mathfrak{n}$.

Now for a fixed $\mathfrak{d}$ divisor of $\mathfrak{n}$, and the corresponding a with $\langle a\rangle+\mathfrak{n}=\mathfrak{d}$, we will consider M-symbols of the form $(a:b)$ under M-symbol equivalence and under the action of $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$.

Let $(a:b)$ be an M-symbol with $\langle a\rangle+\mathfrak{n}=\mathfrak{d}$, $\mathfrak{d}'$ coprime to $\mathfrak{n}$ such that $\mathfrak{d}\mathfrak{d}'\mathfrak{a}=\langle a\rangle$

and $a\mathfrak{a}^{-1} + b\mathfrak{b}^{-1} + \mathfrak{n} = R$. Such an M-symbol satisfies

$$\langle b \rangle + \mathfrak{d} + \mathfrak{n}/\mathfrak{d} = b\mathfrak{b}^{-1} + a\mathfrak{a}^{-1} + \mathfrak{n}/\mathfrak{d} = R,$$

and also

$$(a : b) = (a : b') \Longleftrightarrow b \equiv b' \pmod{\mathfrak{n}/\mathfrak{d}}, \tag{1.2.2}$$

since

$$(a : b) = (a : b') \Leftrightarrow ab \equiv ab' \pmod{\mathfrak{n}} \Leftrightarrow \mathfrak{n} | a(b - b') = \mathfrak{d}\mathfrak{d}'\mathfrak{a}(b - b') \Leftrightarrow \mathfrak{n} | \mathfrak{d}(b - b').$$

We also want to characterise the action of $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$ on our M-symbols $(a : b)$. Instead of doing this directly, we will first study the simpler action of $\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}}$. Recall from §1.2.2 that

$$\Gamma_1^{\mathfrak{a},\mathfrak{b}} = \left\{ \begin{pmatrix} 1 & y \\ 0 & w \end{pmatrix} | \, y \in \mathfrak{a}^{-1}\mathfrak{b},\, w \in R^\times \right\} \supseteq \Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}} = \left\{ \begin{pmatrix} 1 & y \\ 0 & 1 \end{pmatrix} | \, y \in \mathfrak{a}^{-1}\mathfrak{b} \right\}.$$

For the smaller group:

$$(a : b) \longmapsto (a : b') = (a : b) \begin{pmatrix} 1 & y \\ 0 & 1 \end{pmatrix} = (a : ay + b),$$

so that under $\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}}$-action we have

$$(a : b) \sim (a : b') \Longleftrightarrow b - b' \in a\mathfrak{a}^{-1}\mathfrak{b} = \mathfrak{d}\mathfrak{d}'\mathfrak{b}.$$

From this and the previous M-symbol equivalence condition (1.2.2), we see that we can identify $(a : b)$ and $(a : b')$ whenever $b \equiv b' \pmod{\mathfrak{n}/\mathfrak{d} + \mathfrak{d}\mathfrak{d}'\mathfrak{b}}$. Note that, since $\mathfrak{d}'\mathfrak{b}$ is coprime to $\mathfrak{n}$, $\mathfrak{n}/\mathfrak{d} + \mathfrak{d}\mathfrak{d}'\mathfrak{b} = \mathfrak{n}/\mathfrak{d} + \mathfrak{d}$. So for our fixed class $[\mathfrak{a}]$ and each $\mathfrak{d}$ divisor of $\mathfrak{n}$, there are $\varphi(\mathfrak{d} + \mathfrak{n}/\mathfrak{d})$ orbits of M-symbols of type $(\mathfrak{a}, \mathfrak{b})$ (under $\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}}$-action).

Now we go back to the group $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$. Its action on M-symbols is given by:

$$(a : b) \longmapsto (a : b') = (a : b) \begin{pmatrix} 1 & y \\ 0 & w \end{pmatrix} = (a : ay + bw),$$

with $y \in \mathfrak{a}^{-1}\mathfrak{b}$, $w \in R^\times$. That is, two M-symbols $(a : b)$ and $(a : b')$ are $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$-equivalent if and only if $b - ub' \in a\mathfrak{a}^{-1}\mathfrak{b} = \mathfrak{d}\mathfrak{d}'\mathfrak{b}$, for some $u \in R^\times$. In particular, it is clear that we can identify $(a : b)$ and $(a : b')$ under $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$-action if and only if there

exists $u \in R^\times$ such that $(a : b) \sim (a : ub')$ under $\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}}$-action. With our previous count of $\Gamma_{1,1}^{\mathfrak{a},\mathfrak{b}}$-equivalence classes, we now deduce that for each $\mathfrak{d}$ divisor of $\mathfrak{n}$ we have $\varphi_{\mathrm{u}}(\mathfrak{d} + \mathfrak{n}/\mathfrak{d})$ orbits of M-symbols under $\Gamma_1^{\mathfrak{a},\mathfrak{b}}$-action. Thus we have proved that the number of $\Gamma_0(\mathfrak{n})$-orbits of cusps in $[\mathfrak{a}]$ is:

$$\sum_{\mathfrak{d}|\mathfrak{n}} \varphi_{\mathrm{u}}(\mathfrak{d} + \mathfrak{n}/\mathfrak{d}).$$

□

Enumerating equivalence classes of cusps and principal M-symbols

Following the proof of Proposition 1.2.29, we can write algorithms to compute a list of representatives for $\Gamma_0(\mathfrak{n})$-orbits of cusps and to enumerate principal M-symbols.

Algorithm 1.2.30 (Obtaining a set of representatives for $\Gamma_0(\mathfrak{n})$-equivalence classes.)**.** *Given a level* $\mathfrak{n}$*, this algorithm computes a list of coset representatives of equivalence classes of cusps under the action of* $\Gamma_0(\mathfrak{n})$*.*

1. Compute a list of representatives $A = \{\mathfrak{a}_1, \ldots, \mathfrak{a}_h\}$, with $\mathfrak{a}_i$ coprime to $\mathfrak{n}$, for the ideal classes in K.

2. For each $\mathfrak{a} \in A$, fix $\mathfrak{b}$ in the inverse class to $\mathfrak{a}$, coprime to $\mathfrak{n}\mathfrak{a}$. We will now loop over the divisors of $\mathfrak{n}$.

   For every $\mathfrak{d}|\mathfrak{n}$:

   (a) Find $\mathfrak{d}'$ coprime to $\mathfrak{n}\mathfrak{b}$ in inverse class to $\mathfrak{d}\mathfrak{a}$.

   (b) Find a such that $\mathfrak{d}'\mathfrak{d}\mathfrak{a} = \langle a \rangle$.

   (c) Loop through representatives of cosets in $\left(R/\left(\mathfrak{d} + \mathfrak{n}/\mathfrak{d}\right)\right)^\times / U_{\mathfrak{d}+\mathfrak{n}/\mathfrak{d}}$.

   For each representative x:

   (i) Lift x to a solution b coprime to a and such that $b \in \mathfrak{b}$:

   $$\begin{array}{rcl} (R/\langle a \rangle)^\times & \longrightarrow & \left(R/\left(\mathfrak{d} + \mathfrak{n}\mathfrak{d}^{-1}\right)\right)^\times / U_{\mathfrak{d}+\mathfrak{n}/\mathfrak{d}} \\ b & \longmapsto & x. \end{array}$$

   (ii) Complete the pair (a, b) to an $(\mathfrak{a}, \mathfrak{b})$-matrix $\begin{pmatrix} a' & b' \\ a & b \end{pmatrix}$.

   (iii) Add the cusp a'/a to the list of cusp representatives.

Example 1.2.31. We have implemented Algorithm 1.2.30 in *Sage*. Let us consider a simple case, $k = \mathbb{Q}(\sqrt{-5})$ and $\mathfrak{n} = \langle 3 \rangle$. We then can compute a list of representatives for $\Gamma_0(\mathfrak{n})$-orbits of cusps with the function `Gamma0_NFCusps`:

```
sage: k.<a> = NumberField(x^2 + 5)
sage: N = k.ideal(3)

sage: Gamma0_NFCusps(N)
[Cusp [0: 1] of Number Field in a with defining polynomial x^2 + 5,
Cusp [1: 3] of Number Field in a with defining polynomial x^2 + 5,
Cusp [1: -a + 8] of Number Field in a with defining polynomial x^2 + 5,
...
Cusp [73*a - 170: 2*a + 1] of Number Field ...]
```

We can also use *Sage* to verify that our representatives are not equivalent under $\Gamma_0(\mathfrak{n})$-action:

```
sage: L = Gamma0_NFCusps(N)
sage: all([not L[i].is_Gamma0_equivalent(L[j], N) for i, j in \
                                    mrange([len(L), len(L)]) if i<j])
True
```

The function `number_of_Gamma0_NFCusps` computes the formula of Proposition 1.2.29, so it is possible to check that we have obtained the right number of orbits.

```
sage: from sage.modular.cusps_nf import number_of_Gamma0_NFCusps
sage: number_of_Gamma0_NFCusps(N)
8
sage: len(Gamma0_NFCusps(N))
8
```

Now recall that principal M-symbols of level $\mathfrak{n}$ (or M-symbols of type (R, R) and level $\mathfrak{n}$) are coprime pairs $(c, d) \in R \oplus R$ modulo the equivalence relation:

$$(c, d) \sim (c', d') \iff cd' \equiv c'd \pmod{\mathfrak{n}}.$$

That is, the set of principal M-symbols is $\mathbb{P}^1(R/\mathfrak{n})$.

In the proof of Proposition 1.2.29 we have seen that to any principal M-symbol $(c : d)$ we can associate a divisor of the level $\mathfrak{n}$, $\mathfrak{d} = \langle c \rangle + \mathfrak{n}$, and conversely, the choice of a divisor $\mathfrak{d}|\mathfrak{n}$ determines the first coefficient of the M-symbol. We can use this information to formulate the following algorithm.

Algorithm 1.2.32 (Enumerating principal M-symbols of level $\mathfrak{n}$)**.** *Given an ideal* $\mathfrak{n}$*, this algorithm enumerates all principal M-symbols of level* $\mathfrak{n}$*, that is, a list of representatives of all the elements of* $\mathbb{P}^1(R/\mathfrak{n})$*.*

For each divisor $\mathfrak{d}|\mathfrak{n}$:

1. Find $\mathfrak{d}'$ coprime to $\mathfrak{n}$ in the inverse class to $\mathfrak{d}$.
2. Find $c \in R$ such that $\langle c \rangle = \mathfrak{d}\mathfrak{d}'$.
3. Define $I = \mathfrak{d} + \mathfrak{n}/\mathfrak{d}$.
4. Loop through a set of representatives of cosets in $R/(\mathfrak{n}/\mathfrak{d})$.
   For each representative d:
   If $\langle d \rangle + I = R$ then
   (a) find d' such that
   $$d' \equiv d \pmod{\mathfrak{n}/\mathfrak{d}},$$
   $$\langle c \rangle + \langle d' \rangle = R.$$
   (b) Add the $(c : d')$ to the list of principal M-symbols.

Example 1.2.33. We can see Algorithm 1.2.32 in action in the following example, where we use our implementation of lists of (principal) M-symbols in *sage*.

The function `P1NFList` creates the set $\mathbb{P}^1(R/\mathfrak{n})$ for a given ideal $\mathfrak{n}$ of a number field. For instance:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a^2 - a + 1)

sage: P = P1NFList(N); P
The projective line over the ring of
integers modulo the Fractional ideal (5, a^2 - a + 1)
```

We can list the elements of the set returned by `P1NFList`, and using the function `psi`, which computes the index $[\Gamma : \Gamma_0(\mathfrak{n})]$, it is possible to check that our list has the right number of elements (recall Proposition 1.2.21).

```
sage: P.list()
[M-symbol (0: 1) of level Fractional ideal (5, a^2 - a + 1),
M-symbol (1: -2*a^2 - 2*a) of level Fractional ideal (5, a^2 - a + 1),
```

```
M-symbol (1: -a^2 - 2*a) of level Fractional ideal (5, a^2 - a + 1),
...
M-symbol (1: 2*a^2 + 2*a) of level Fractional ideal (5, a^2 - a + 1)]

sage: from sage.modular.modsym.p1list_nf import psi
sage: psi(N)
26
sage: len(P)==psi(N)
True
```

1.3 M-symbols for $\Gamma_1(\mathfrak{n})$

We can adapt our results about M-symbols and cusp equivalence for the congruence subgroup $\Gamma_0(\mathfrak{n})$ to valid results for the congruence subgroup $\Gamma_1(\mathfrak{n})$. We begin again with a quick overview of the situation for $K = \mathbb{Q}$ in §1.3.1, and then look at the general number field case in §1.3.2.

1.3.1 M-symbols for $\Gamma_1(N)$

More details for the material in this section can be found in [12] and [30, Chapter 8].

As in §1.1, take $\Gamma = \mathrm{SL}(2, \mathbb{Z})$. We now consider the congruence subgroup $\Gamma_1(N)$, which is defined for a positive integer N by

$$\Gamma_1(N) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma : \ c \equiv 0, \ a \equiv d \equiv 1 \ (\mathrm{mod}\, N) \right\}.$$

As before, $\Gamma_1(N)$ acts on the set of cusps $\mathbb{P}^1(\mathbb{Q})$ by linear fractional transformations. We can adapt the definition of our M-symbols (recall Definition 1.1.4) to act as coset representatives for $\Gamma_1(N)$ in Γ.

Proposition 1.3.1. *Given two matrices* $M_1 = \begin{pmatrix} a_1 & b_1 \\ c_1 & d_1 \end{pmatrix}$ *and* $M_2 = \begin{pmatrix} a_2 & b_2 \\ c_2 & d_2 \end{pmatrix}$ *in* Γ*, the following are equivalent:*

(i) $M_2 = \gamma M_1$ *for some* $\gamma \in \Gamma_1(N)$,

(ii) $c_1 \equiv c_2 \,(\mathrm{mod}\, N)$ *and* $d_1 \equiv d_2 \,(\mathrm{mod}\, N)$.

Proof. See [30, Proposition 8.6]. □

It is clear then that each right coset in $\Gamma_1(N)\backslash\Gamma$ can be represented by a pair (c,d) with $c,d \in \mathbb{Z}/N\mathbb{Z}$ and $\gcd(c,d,N) = 1$. Thus we define *Manin symbols for* $\Gamma_1(N)$ as these pairs (c,d). The bijection between the Manin symbols and the cosets in $\Gamma_1(N)\backslash\Gamma$ is explicitly given by:

$$(c:d) \quad \leftrightarrow \quad M = \begin{pmatrix} a & b \\ c & d \end{pmatrix},$$

where $a, b \in \mathbb{Z}$ are chosen so that $ad - bc = 1$ and, as in the $\Gamma_0(N)$ case, a different choice of a, b has the effect of multiplying M on the left by a power of T. Note that product by T does not change the right coset of M, since $T \in \Gamma_1(N)$ for all N.

We test the equivalence of cusps under the action of $\Gamma_1(N)$ with the following Lemma.

Lemma 1.3.2. *Let α_1 and α_2 be cusps with representatives p_1/q_1 and p_2/q_2. The following are equivalent:*

(i) $\alpha_2 = \gamma(\alpha_1)$ *for some* $\gamma \in \Gamma_1(N)$,

(ii) $q_2 \equiv q_1 \pmod{N}$ *and* $p_2 \equiv p_1 \pmod{\gcd(q_1, N)}$.

Proof. The proof is copied from [12, Lemma 3.2], where the work is done for the analogue of $\Gamma_1(N)$ in $\mathrm{PSL}_2(\mathbb{Z})$. The congruences in (ii) follow directly if we assume (i).

For the converse we use Proposition 1.3.1. Assume that (ii) holds and write $p_1 s_1' - q_1 r_1' = 1$ and $p_2 s_2 - q_2 r_2 = 1$, with $r_1', s_1', r_2, s_2 \in \mathbb{Z}$.

Since $q_2 \equiv q_1 \pmod{N}$, we have $\gcd(q_1, N) = \gcd(q_2, N) = N_0$. On the other hand, by hypothesis $p_2 \equiv p_1 \pmod{N_0}$, which implies that $s_1' \equiv s_2 \pmod{N_0}$. Hence we may choose $x \in \mathbb{Z}$ such that $xq_1 \equiv s_1' - s_2 \pmod{N}$. Now set

$$s_1 = s_1' - xq_1 \quad \text{and} \quad r_1 = r_1' - xq_1,$$

and we are in the conditions of Proposition 1.3.1. Namely $p_1 s_1 - q_1 r_1 = 1$, $q_1 \equiv q_2 \pmod{N}$ and $s_1 \equiv s_2 \pmod{N}$. Thus we know there exists $\gamma \in \Gamma_1(N)$ such that

$$\begin{pmatrix} p_2 & r_2 \\ q_2 & s_2 \end{pmatrix} = \gamma \begin{pmatrix} p_1 & r_1 \\ q_1 & s_1 \end{pmatrix},$$

and it follows that $p_2/q_2 = \gamma(p_1/q_1)$. □

1.3.2 M-symbols for $\Gamma_1(\mathfrak{n})$

We now return to the setup of §1.2. Let K be a number field with ring of integers R and class number h_K. As before we define $\Gamma = \mathrm{GL}(2, R)$. For a level $\mathfrak{n}$ consider the congruence subgroup $\Gamma_1(\mathfrak{n})$, which we define as follows:

$$\Gamma_1(\mathfrak{n}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma \,|\, c,\ d-1 \in \mathfrak{n} \right\}.$$

We will now rewrite the main results of §1.2 for $\Gamma_1(\mathfrak{n})$.

$(\mathfrak{a}, \mathfrak{b})$-matrices under the action of $\Gamma_1(\mathfrak{n})$

As we did in the previous section for $\Gamma_0(\mathfrak{n})$, we now study the action by left multiplication of $\Gamma_1(\mathfrak{n})$ on the set of $(\mathfrak{a}, \mathfrak{b})$-matrices. The results we obtain will be used in our definition of M-symbols for $\Gamma_1(\mathfrak{n})$ and in the study of cusp equivalence under $\Gamma_1(\mathfrak{n})$.

Proposition 1.3.3. *Let $\mathfrak{a}, \mathfrak{b}$ be ideals in inverse classes, and let*

$$M_1 = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix}, \quad M_2 = \begin{pmatrix} a_1' & b_1' \\ a_2' & b_2' \end{pmatrix},$$

be any two $(\mathfrak{a}, \mathfrak{b})$-matrices. The following statements are equivalent:

(i) *$M_2 = \gamma M_1$ with $\gamma \in \Gamma_1(\mathfrak{n})$.*

(ii) *The following congruences hold:*

  (a) $a_2' b_2 \equiv a_2 b_2' \pmod{\mathfrak{abn}}$,
  (b) $a_2 \equiv a_2' \pmod{\mathfrak{n}}$,
  (c) $b_2 \equiv b_2' \pmod{\mathfrak{n}}$.

(iii) *There exists $u \in R$ with $u - 1 \in \mathfrak{n}$ such that:*

  (a) $a_2' = u a_2 \pmod{\mathfrak{an}}$,
  (b) $b_2' = u b_2 \pmod{\mathfrak{bn}}$.

Proof. We follow the proof of the corresponding result for $\Gamma_0(\mathfrak{n})$, Proposition 1.2.13.

Let $g = \det M_1$, so $\langle g \rangle = \mathfrak{ab}$, and $\det M_2 = g u_0$, with $u_0 \in R^\times$. Take $\gamma = M_2 M_1^{-1}$, so that $\det \gamma = u_0$ and $\gamma \in \Gamma = \mathrm{GL}(2, R)$. Now write

$$\gamma = M_2 M_1^{-1} = \begin{pmatrix} v & x \\ y & u \end{pmatrix}.$$

Clearly $\gamma \in \Gamma_1(\mathfrak{n})$ if and only if the following hold:

$$\begin{aligned} y &= g^{-1}(a_2'b_2 - a_2b_2') \in \mathfrak{n}, \\ u-1 &= g^{-1}(a_1b_2' - b_1a_2') - 1 \in \mathfrak{n}. \end{aligned} \tag{1.3.1}$$

Now, it is easy to check that if the conditions of (1.3.1) above are satisfied, the congruences in (ii) follow. We obtain (a) directly from the first condition:

$$g^{-1}(a_2'b_2 - a_2b_2') \in \mathfrak{n} \Leftrightarrow a_2'b_2 \equiv a_2b_2' \pmod{\mathfrak{abn}}.$$

With respect to (b) and (c), observe that:

$$\begin{aligned} g^{-1}(a_1b_2' - b_1a_2') - 1 \in \mathfrak{n} &\Rightarrow g^{-1}(a_1b_2'a_2 - b_1a_2'a_2) - a_2 \in \mathfrak{n} \\ &\Rightarrow g^{-1}(a_1b_2a_2' - b_1a_2'a_2) - a_2 \in \mathfrak{n} \\ &\Rightarrow a_2'g^{-1}(a_1b_2 - b_1a_2) - a_2 \in \mathfrak{n} \\ &\Rightarrow a_2' - a_2 \in \mathfrak{n}, \end{aligned}$$

and repeating the same argument for b_2 we obtain $b_2' \equiv b_2 \pmod{\mathfrak{n}}$.

Conversely, if the congruences in (ii) hold, then from $a_2 \equiv a_2'$ and $b_2 \equiv b_2' \pmod{\mathfrak{n}}$ it follows that $g^{-1}(a_2b_2' - b_1a_2') \equiv 1 \pmod{\mathfrak{n}}$. Thus both conditions in (1.3.1) are satisfied and $\gamma \in \Gamma_1(\mathfrak{n})$.

Now we assume that (i) and (ii) are true. Part (iii) follows immediately, since:

$$\begin{aligned} a_2' &= ya_1 + ua_2 \equiv ua_2 \pmod{\mathfrak{an}}, \\ b_2' &= yb_1 + ub_2 \equiv ub_2 \pmod{\mathfrak{bn}}, \end{aligned}$$

where $u - 1 \in \mathfrak{n}$ since $\gamma \in \Gamma_1(\mathfrak{n})$.

Finally we assume that the congruences in (iii) hold. Since $u - 1 \in \mathfrak{n}$, it follows that $a_2' \equiv a_2 \pmod{\mathfrak{n}}$ and $b_2' \equiv b_2 \pmod{\mathfrak{n}}$. Now for congruence (a), observe that

$$ua_2b_2' \equiv a_2'b_2' \equiv ub_2a_2' \pmod{\mathfrak{abn}},$$

and in particular, $ua_2b_2'g^{-1} \equiv ub_2a_2'g^{-1} \pmod{\mathfrak{n}}$. Now recall that $u - 1 \in \mathfrak{n}$, and multiplying again by g on both sides we obtain (ii)(a).

□

Proposition 1.3.4. *If any of the equivalent statements of the previous result holds, then there exist* $u_0 \in R^\times$, $u \equiv 1 \pmod{\mathfrak{n}}$ *and* $\mathfrak{d}$ *a divisor of* $\mathfrak{n}$ *such that:*

(i) $\langle a_2 \rangle + \mathfrak{an} = \langle a_2' \rangle + \mathfrak{an} = \mathfrak{ad}$,

(ii) $a_2' \equiv u a_2 \pmod{\mathfrak{an}}$,

(iii) $u a_1' \equiv u_0 a_1 \pmod{\mathfrak{ad}}$.

Conversely, if these conditions hold, there exists $\gamma \in \Gamma_1(\mathfrak{n})$ *such that*

$$\gamma M_1 = M_2' = M_2 \begin{pmatrix} 1 & w \\ 0 & 1 \end{pmatrix}, \quad \textit{with } w \in \mathfrak{a}^{-1}\mathfrak{b}.$$

Proof. It is a copy of the proof of Proposition 1.2.14, so we leave out the details. □

Manin symbols for $\Gamma_1(\mathfrak{n})$

Now we adjust our definition of M-symbols so that they serve as coset representatives for $\Gamma_1(\mathfrak{n})\backslash\Gamma$. Taking into account Proposition 1.3.3, we make the following definition.

Definition 1.3.5. An *M-symbol of level* $\mathfrak{n}$ *and type* $(\mathfrak{a}, \mathfrak{b})$ *for* $\Gamma_1(\mathfrak{n})$ is an equivalence class of

$$\left\{(a, b) \in \mathfrak{a} \oplus \mathfrak{b} : a\mathfrak{a}^{-1} + b\mathfrak{b}^{-1} = R\right\} / \sim,$$

where:

$$\begin{aligned} (a, b) \sim (a', b') &\iff ab' \equiv a'b \pmod{\mathfrak{abn}} \text{ and } \begin{cases} a \equiv a' \pmod{\mathfrak{an}} \\ b \equiv b' \pmod{\mathfrak{bn}} \end{cases} \\ &\iff \text{there exists } u \in R \text{ such that } u - 1 \in \mathfrak{n} \text{ and} \\ &\qquad \begin{cases} ua \equiv a' \pmod{\mathfrak{an}} \\ ub \equiv b' \pmod{\mathfrak{bn}}. \end{cases} \end{aligned}$$

We refer to M-symbols of type (R, R) and level $\mathfrak{n}$ as *principal M-symbols of level* $\mathfrak{n}$ *for* $\Gamma_1(\mathfrak{n})$.

Note that principal M-symbols are a direct generalization of the M-symbols for $\Gamma_1(N)$ that we defined for $K = \mathbb{Q}$ (§1.3.1).

Remark 1.3.6. As we did for $\Gamma_0(\mathfrak{n})$, we may simplify the equivalence relations defining M-symbols if we choose our ideals $\mathfrak{a}, \mathfrak{b}$ so that $\mathfrak{ab}$ is coprime to $\mathfrak{n}$. However, in this case we do not obtain two simplified equivalent sets of relations as it happened

in Remark 1.2.20. With $\mathfrak{ab}$ coprime to $\mathfrak{n}$, the two equivalent sets of conditions in Definition 1.3.5 reduce to the same equations, namely:

$$(a,b) \sim (a',b') \iff \begin{cases} a \equiv a' \pmod{\mathfrak{n}} \\ b \equiv b' \pmod{\mathfrak{n}}. \end{cases}$$

It is clear from our definition that we have a bijection between M-symbols of type $(\mathfrak{a},\mathfrak{b})$ and level $\mathfrak{n}$ for $\Gamma_1(\mathfrak{n})$ and orbits of $(\mathfrak{a},\mathfrak{b})$-matrices under the action of $\Gamma_1(\mathfrak{n})$:

$$\{\text{M-symbols of type } (\mathfrak{a},\mathfrak{b}) \text{ for } \Gamma_1(\mathfrak{n})\} \longleftrightarrow \{\text{Set of orbits of } \Gamma_1(\mathfrak{n})\backslash X_{\mathfrak{a},\mathfrak{b}}\}.$$

We also have an analogue to Proposition 1.2.21:

Proposition 1.3.7. *For every pair of ideals $\mathfrak{a},\mathfrak{b}$ in inverse classes, the number of M-symbols of level $\mathfrak{n}$ and type $(\mathfrak{a},\mathfrak{b})$ for $\Gamma_1(\mathfrak{n})$ is $[\Gamma : \Gamma_1(\mathfrak{n})]$.*

Proof. We need only to copy the proof of Proposition 1.2.21 replacing $\Gamma_0(\mathfrak{n})$ by $\Gamma_1(\mathfrak{n})$. □

Cusp equivalence under $\Gamma_1(\mathfrak{n})$-action

Let α,α' be two cusps in the same ideal class, and choose representatives $\alpha = a_1/a_2$ and $\alpha' = a_1'/a_2'$ with the same ideal $\mathfrak{a} = \langle a_1,a_2\rangle = \langle a_1',a_2'\rangle$. One of the key points in our study of $\Gamma_0(\mathfrak{n})$-orbits of cusps in §1.2.6 was the equivalence between the existence of an element $\gamma \in \Gamma_0(\mathfrak{n})$ with $\gamma(\alpha) = \alpha'$ and the existence of $\gamma \in \Gamma_0(\mathfrak{n})$ such that $\gamma \binom{a_1}{a_2} = \binom{a_1'}{a_2'}$. This equivalence holds because any matrix of the form wI, where $w \in R^\times$ and I is the 2×2 identity matrix, is in $\Gamma_0(\mathfrak{n})$. This is not true for $\Gamma_1(\mathfrak{n})$ in general (it holds only if $w-1 \in \mathfrak{n}$). Instead we have the following:

Lemma 1.3.8. *Let α,α' be two cusps in the same ideal class. Choose representatives $\alpha = a_1/a_2$ and $\alpha' = a_1'/a_2'$ with the same ideal $\mathfrak{a} = \langle a_1,a_2\rangle = \langle a_1',a_2'\rangle$. Then the following are equivalent:*

(i) *There exists $\gamma \in \Gamma_1(\mathfrak{n})$ such that $\gamma\alpha = \alpha'$.*

(ii) *There exist $\gamma \in \Gamma_1(\mathfrak{n})$ and $w_0 \in R^\times$ such that*

$$\gamma \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} a_1' \\ w_0 a_2' \end{pmatrix}.$$

Proof. Let α and α' be $\Gamma_1(\mathfrak{n})$-equivalent, so that there exists $\gamma \in \Gamma_1(\mathfrak{n})$ such that $\gamma\alpha = \alpha'$. We then know that:

$$\gamma \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} w_0 a_1' \\ w_0 a_2' \end{pmatrix} = w_0 \begin{pmatrix} a_1' \\ a_2' \end{pmatrix},$$

for some $w_0 \in R^\times$ (recall Proposition 1.2.23). Now if we take $\gamma_1 = \begin{pmatrix} w_0^{-1} & 0 \\ 0 & 1 \end{pmatrix} \gamma$, we have:

$$\gamma_1 \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} a_1' \\ w_0 a_2' \end{pmatrix},$$

and clearly $\gamma_1 \in \Gamma_1(\mathfrak{n})$ since $w_0 \in R^\times$.

Now assume that (ii) holds. Set $\gamma_1 = \begin{pmatrix} w_0 & 0 \\ 0 & 1 \end{pmatrix} \gamma$ and then $\gamma_1\alpha = \alpha'$ with $\gamma_1 \in \Gamma_1(\mathfrak{n})$. □

Taking into account Lemma 1.3.8 and following the strategies in §1.2.6 we find the following test for $\Gamma_1(\mathfrak{n})$-equivalence of cusps, which generalizes the result for $K = \mathbb{Q}$, Lemma 1.3.2.

Proposition 1.3.9. *Let α, α' be two cusps in the same ideal class. Choose representatives $\alpha = a_1/a_2$ and $\alpha' = a_1'/a_2'$ with the same ideal $\mathfrak{a} = \langle a_1, a_2 \rangle = \langle a_1', a_2' \rangle$. Then the following are equivalent:*

(i) *α and α' are $\Gamma_1(\mathfrak{n})$-equivalent,*

(ii) *there exist $u \in R$ such that $u - 1 \in \mathfrak{n}$, $u_0, w_0 \in R^\times$ and a divisor $\mathfrak{d}$ of $\mathfrak{n}$ such that:*

  (a) $\mathfrak{d}_\mathfrak{n}(\alpha) = \mathfrak{d}_\mathfrak{n}(\alpha') = \mathfrak{d}$,

  (b) $w_0 a_2' \equiv u a_2 \pmod{\mathfrak{a}\mathfrak{n}}$,

  (c) $u a_1' \equiv u_0 a_1 \pmod{\mathfrak{a}\mathfrak{d}}$.

In case $\mathfrak{a}$ and $\mathfrak{n}$ are coprime, we can replace (ii) *by the simpler:*

(ii)′ *there exist $u_0, w_0 \in R^\times$ and a divisor $\mathfrak{d}$ of $\mathfrak{n}$ such that:*

  (a) $\langle a_2 \rangle + \mathfrak{n} = \langle a_2' \rangle + \mathfrak{n} = \mathfrak{d}$,

  (b) $w_0 a_2' \equiv a_2 \pmod{\mathfrak{n}}$,

  (c) $a_1' \equiv u_0 a_1 \pmod{\mathfrak{d}}$.

Proof. We follow the proof for Proposition 1.2.26.

Suppose that α and α' are $\Gamma_1(\mathfrak{n})$-equivalent. By Lemma 1.3.8 we know that there exist a matrix $\gamma \in \Gamma_1(\mathfrak{n})$ and an element $w_0 \in R^\times$ such that $\gamma \left(\begin{smallmatrix} a_1 \\ a_2 \end{smallmatrix}\right) = \left(\begin{smallmatrix} a_1' \\ w_0 a_2' \end{smallmatrix}\right)$. Complete the pair $\left(\begin{smallmatrix} a_1 \\ a_2 \end{smallmatrix}\right)$ to an $(\mathfrak{a}, \mathfrak{b})$-matrix M_1 and define $M_2 = \gamma M_1$. Then M_2 is an $(\mathfrak{a}, \mathfrak{b})$-matrix with first column $\left(\begin{smallmatrix} a_1' \\ w_0 a_2' \end{smallmatrix}\right)$ and (ii) follows from Proposition 1.3.4.

Conversely, assume (ii) holds. We are again in the conditions of Proposition 1.3.4. Hence there exists $\gamma \in \Gamma_1(\mathfrak{n})$ such that $\gamma M_1 = M_2'$, where M_2' is another $(\mathfrak{a}, \mathfrak{b})$-matrix with the same first column and determinant as M_2 above. In particular, this means that $\gamma \left(\begin{smallmatrix} a_1 \\ a_2 \end{smallmatrix}\right) = \left(\begin{smallmatrix} a_1' \\ w_0 a_2' \end{smallmatrix}\right)$, which by Lemma 1.3.8 is equivalent to (i).

For the last part, recall that $\mathfrak{d}_\mathfrak{n}(\alpha) = \langle a_2 \rangle \mathfrak{a}^{-1} + \mathfrak{n}$ and $\langle a_2 \rangle \mathfrak{a}^{-1} + \mathfrak{n} = \langle a_2 \rangle + \mathfrak{n}$ if $\mathfrak{a}$ is coprime to $\mathfrak{n}$, and for the same reason $\mathfrak{d}_\mathfrak{n}(\alpha') = \langle a_2' \rangle + \mathfrak{n}$. It is also clear that when $\mathfrak{a}$ and $\mathfrak{n}$ are coprime, for $u \equiv 1 \pmod{\mathfrak{n}}$ we have

$$w_0 a_2' - a_2 \in \mathfrak{n} \Leftrightarrow w_0 a_2' - u a_2 \in \mathfrak{a}\mathfrak{n},$$

and similarly

$$a_1' - u_0 a_1 \in \mathfrak{d} \Leftrightarrow u a_1' - u_0 a_1 \in \mathfrak{d}\mathfrak{a}.$$

□

Chapter 2

Normaliser groups

In this chapter we study some elements which normalise the congruence subgroup $\Gamma_0(\mathfrak{n})$. For a number field K with ring of integers R we consider the normaliser Δ of $\Gamma = \mathrm{GL}(2,R)$ in $\mathrm{GL}(2,K)$ and its subgroup $\Delta_0(\mathfrak{n})$. As shown in J. Bygott's thesis [6], in the imaginary quadratic case the introduction of the normaliser group Δ leads to simplified geometry in the hyperbolic space (we discuss this at more length in §3.5). In §2.1 we review previous results on the structure of this group Δ. Using our work for $(\mathfrak{a},\mathfrak{b})$-matrices in Chapter 1 we rewrite and complete some of this theory, obtaining new results on $\Delta_0(\mathfrak{n})$-equivalence of cusps.

In §2.2 we recall some classical theory about the normaliser of $\Gamma_0(N)$. This normaliser group was first described by Newman and Lehner [21]. We give a detailed proof of their Theorem, which is slightly different from the one in [21] and somewhat clearer. Then we proceed to generalise some of this theory to the number field case. We describe Atkin-Lehner type involutions over number fields and our conjectures on the normaliser group of $\Gamma_0(\mathfrak{n})$.

2.1 The groups Δ and $\Delta_0(\mathfrak{n})$

Let R be a Dedekind domain, K its field of fractions and Cl the ideal class group. Take $G = \mathrm{GL}(n,K)$ and $\Gamma = \mathrm{GL}(n,R)$ the groups of invertible $n \times n$ matrices over K and R. In [11] we find a description of the structure of $\Delta = \mathcal{N}_G(\Gamma)$, the normaliser of Γ in G. We now recall the main result in [11], which describes this normaliser group Δ for $n \in \mathbb{N}$ (although here we are only interested in $n = 2$).

Theorem 2.1.1. *With notation as above, let* $\mathrm{Cl}[n] = \{c \in \mathrm{Cl} : c^n = 1\}$. *Then:*

(i) *For a matrix* $M \in G$,

$$M \in \Delta \Leftrightarrow \langle M \rangle^n = \langle \det M \rangle,$$

where $\langle M \rangle$ *denotes the fractional ideal generated by the entries of* M.

(ii) *There exists a group isomorphism*

$$\Delta / K^\times \Gamma \cong \mathrm{Cl}[n],$$

where we identify $K^\times$ *with the centre of* G, *given by the scalar matrices* $\{\lambda I_n | \, \lambda \in K^\times\}$.

Proof. [11, Theorem 1] □

Let us observe now that for $M \in \Delta$, the ideal $\langle M \rangle$ is in fact generated by the entries in any one row or column of M (see [11, Remark 2] or [6, Lemma 31] for a proof). In fact, if we denote $\mathfrak{a} = \langle M \rangle$, we have an isomorphism

$$\begin{array}{ccc} \overbrace{R \oplus \cdots \oplus R}^{n} & \to & \overbrace{\mathfrak{a} \oplus \cdots \oplus \mathfrak{a}}^{n}. \\ v & \mapsto & vM \end{array}$$

From now on we restrict to the case $n = 2$, i.e. we now have $G = \mathrm{GL}(2, K)$ and $\Gamma = \mathrm{GL}(2, R)$. From the theorem above we know that $\Delta / K^\times \Gamma \cong \mathrm{Cl}[2]$. In particular, we obtain the following description of Δ in terms of $(\mathfrak{a}, \mathfrak{b})$-matrices.

Lemma 2.1.2. *The normaliser group* Δ *can be described as follows:*

$$\Delta = \left\{ M : M \in X_{\mathfrak{a},\mathfrak{a}} \text{ for any } \mathfrak{a} \subseteq R \text{ such that } [\mathfrak{a}^2] = 1 \right\}.$$

Proof. Let $M \in \Delta$. Then for $\mathfrak{a} = \langle M \rangle$ we have $(R \oplus R)M = \mathfrak{a} \oplus \mathfrak{a}$, so M is an $(\mathfrak{a}, \mathfrak{a})$-matrix. Conversely, any $(\mathfrak{a}, \mathfrak{a})$-matrix M satisfies $\langle \det M \rangle = \mathfrak{a}^2 = \langle M \rangle^2$, so $M \in \Delta$. □

In [6, §1.4], J.Bygott introduces the following subgroup of Δ:

Definition 2.1.3. For any level $\mathfrak{n} \in R$, we define the subgroup $\Delta_0(\mathfrak{n})$ as follows:

$$\Delta_0(\mathfrak{n}) = \left\{ M \in \Delta : \ \exists \lambda \in K^\times \text{ with } \lambda \langle M \rangle + \mathfrak{n} = R \text{ and } \lambda M \equiv \begin{pmatrix} * & * \\ 0 & * \end{pmatrix} \pmod{\mathfrak{n}} \right\}.$$

A more intrinsic definition can be given characterising $\Delta_0(\mathfrak{n})$ as a normaliser group. With this purpose we define, analogously to the rational case, the following subgroups of Γ:

$$\begin{aligned}
\Gamma_1(\mathfrak{n}) &= \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0(\mathfrak{n}) \,|\, d-1 \in \mathfrak{n} \right\}, \\
\Gamma_1^1(\mathfrak{n}) &= \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_1(\mathfrak{n}) \,|\, a-1 \in \mathfrak{n} \right\}, \\
\Gamma(\mathfrak{n}) &= \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_1^1(\mathfrak{n}) \,|\, b \in \mathfrak{n} \right\}.
\end{aligned}$$

Proposition 2.1.4. *The group $\Gamma(\mathfrak{n})$ is normal in Δ. The normaliser of both $\Gamma_1^1(\mathfrak{n})$ and $\Gamma_1(\mathfrak{n})$ in Δ is $\Delta_0(\mathfrak{n})$.*

Proof. See [6, Proposition 33]. □

Remark 2.1.5. Note that the normaliser of $\Gamma_0(\mathfrak{n})$ in Δ can be strictly larger than $\Delta_0(\mathfrak{n})$, as shown in [6, Corollary 34].

We have an analogue of the isomorphism in part (ii) of Theorem 2.1.1:

Proposition 2.1.6. $\Delta_0(\mathfrak{n}) \cap \Gamma = \Gamma_0(\mathfrak{n})$*, and we have an isomorphism*

$$\Delta_0(\mathfrak{n})/K^\times \Gamma_0(\mathfrak{n}) \cong \mathrm{Cl}[2].$$

Proof. See [6, Proposition 35 and Corollary 36]. □

However, as we did for Δ, we can describe $\Delta_0(\mathfrak{n})$ in terms of $(\mathfrak{a}, \mathfrak{a})$-matrices. This description will be useful in the next section.

Lemma 2.1.7. *The normaliser group $\Delta_0(\mathfrak{n})$ can be described as follows*

$$\Delta_0(\mathfrak{n}) = \left\{ M : M \text{ is an } (\mathfrak{a}, \mathfrak{a})\text{-matrix of level } \mathfrak{n} \text{ for any } \mathfrak{a} \subseteq R \text{ such that } [\mathfrak{a}^2] = 1 \right\}.$$

Proof. Write $M = \begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} \in \Delta_0(\mathfrak{n})$. In particular, $M \in \Delta$ and therefore M is an $(\mathfrak{a}, \mathfrak{a})$-matrix with $\mathfrak{a} = \langle M \rangle$. Take λ as in Definition 2.1.3. Then $\lambda\mathfrak{a}$ is coprime to $\mathfrak{n}$ and $\lambda a_2 \in \mathfrak{n}$. But since M is an $(\mathfrak{a}, \mathfrak{a})$-matrix, we also have that $\lambda a_2 \in \lambda\mathfrak{a}$. Then $\lambda a_2 \in \lambda\mathfrak{a}\mathfrak{n}$, so that $a_2 \in \mathfrak{a}\mathfrak{n}$ and this proves that M is of level $\mathfrak{n}$.

Conversely, let M be an $(\mathfrak{a}, \mathfrak{a})$-matrix of level $\mathfrak{n}$, with $\mathfrak{a} = \langle M \rangle$ and $a_2 \in \mathfrak{a}\mathfrak{n}$. If $\mathfrak{a}$ is coprime to $\mathfrak{n}$ it is clear that $M \in \Delta_0(\mathfrak{n})$. Otherwise, take $\lambda \in K^\times$ such that $\lambda\mathfrak{a}$ is coprime to $\mathfrak{n}$. Since $\lambda a_2 \in \lambda\mathfrak{a}\mathfrak{n}$, it follows that $\lambda a_2 \in \mathfrak{n}$, and so $M \in \Delta_0(\mathfrak{n})$. □

2.1.1 Cusps and cusp equivalence under Δ and $\Delta_0(\mathfrak{n})$

We will now look at the set of cusps $\mathbb{P}^1(K)$ under the action of the normaliser groups Δ and $\Delta_0(\mathfrak{n})$. Our purpose now is to generalise the results in Chapter 1 concerning cusps and cusp equivalence for the normaliser groups defined in the previous section.

We now recall some basic results about Δ-equivalence of cusps which were proved in J. Bygott's thesis [6].

Lemma 2.1.8. *Let $M \in \Delta$ and $\alpha \in \mathbb{P}^1(K)$. Then $[M\alpha] = [\langle M \rangle][\alpha]$. More precisely, for $a, b \in R$ we have,*

$$M \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a' \\ b' \end{pmatrix},$$

where $\langle a', b' \rangle = \langle M \rangle \langle a, b \rangle$.

Proof. [6, Lemma 40]. □

Corollary 2.1.9. *For α, α' cusps of K, the following are equivalent:*

(i) *α and α' are Δ-equivalent, that is, $\alpha' = M\alpha$ for some $M \in \Delta$,*

(ii) *there exists an integral ideal $\mathfrak{a} \in R$ such that $\mathfrak{a}^2$ is principal and $[\alpha'] = [\mathfrak{a}][\alpha]$.*

Proof. [6, Corollary 41]. □

We can rewrite the last result to give us a practical check on Δ-equivalence of cusps as follows.

Corollary 2.1.10. *Let α, α' be cusps of K. Then α and α' are Δ-equivalent if and only if the square of the ideal $\mathfrak{a} = \langle \alpha' \rangle / \langle \alpha \rangle$ is principal.*

It is also possible to write the corollary as an analogue of Proposition 1.2.24,

Corollary 2.1.11. *There is a bijection:*

$$\begin{array}{rcl} \Delta \backslash \mathbb{P}^1(K) & \longrightarrow & \mathrm{Cl} \,/\, \mathrm{Cl}[2] \\ \alpha & \longmapsto & [\alpha]. \end{array}$$

It follows at once from Corollary 2.1.11 that when $\mathrm{Cl} = \mathrm{Cl}[2]$ all cusps are in the same Δ-orbit. In general, we can characterise cusps in the Δ-orbit of ∞ by generalising the notion of principal cusp (this will be especially useful for some of our work in Chapter 3).

Definition 2.1.12. We say that a cusp $\alpha \in \mathbb{P}^1(K)$ is *semi-principal* if its associated class has order two, i.e. $[\alpha]^2 = 1$.

From the above results on Δ-equivalence of cusps it is clear that a cusp α is semi-principal if and only if $M\alpha = \infty$ for some $M \in \Delta$. Hence a principal cusp is always semi-principal ($\Gamma \subset \Delta$).

Now we will use our descriptions of Δ and $\Delta_0(\mathfrak{n})$ in terms of $(\mathfrak{a}, \mathfrak{a})$-matrices to give a condition for $\Delta_0(\mathfrak{n})$-equivalence of cusps. First, recall that for any level $\mathfrak{n}$ and for any ideal $\mathfrak{a} \in R$ whose square is principal it is always possible to find an $(\mathfrak{a}, \mathfrak{a})$-matrix of level $\mathfrak{n}$ (we may choose as the lower left entry of the matrix any element of $\mathfrak{a}$). In particular this implies that every $\Delta_0(\mathfrak{n})$-orbit of cusps contains all ideal classes in the associated coset of Cl[2] in Cl. More precisely:

Lemma 2.1.13. *Let $\alpha \in \mathbb{P}^1(K)$. For a group G, let $G\alpha$ denote the orbit of the cusp α under the action of G. Then for all $\alpha' \in \Delta\alpha$, there exists a cusp $\alpha'' \in \Delta_0(\mathfrak{n})\alpha$ such that $[\alpha''] = [\alpha']$.*

Proof. Since α' is Δ-equivalent to α, there exists an $(\mathfrak{a}, \mathfrak{a})$-matrix $M \in \Delta$ such that $\alpha' = M\alpha$, where $\langle M \rangle = \mathfrak{a} = \langle \alpha' \rangle / \langle \alpha \rangle$. We can then construct an $(\mathfrak{a}, \mathfrak{a})$-matrix of level $\mathfrak{n}$, M_0, and define $\alpha'' = M_0\alpha$. Clearly α'' verifies the conditions in the lemma:

$$\alpha'' \in \Delta_0(\mathfrak{n})\alpha \quad \text{and} \quad [\alpha''] = [\langle M_0 \rangle][\alpha] = [\mathfrak{a}][\alpha] = [\langle M \rangle][\alpha] = [\alpha'].$$

□

As a consequence, we can now reduce $\Delta_0(\mathfrak{n})$-equivalence of cusps to a check for $\Gamma_0(\mathfrak{n})$-equivalence, for which we found a practical test in Chapter 1 (Corollary 1.2.27).

Proposition 2.1.14. *Let α, $\alpha' \in \mathbb{P}^1(K)$, which are Δ-equivalent. Then the following are equivalent:*

(i) *α, α' are $\Delta_0(\mathfrak{n})$-equivalent,*

(ii) *α'' and α' are $\Gamma_0(\mathfrak{n})$-equivalent, where $\alpha'' \in \Delta_0(\mathfrak{n})\alpha$ is as defined in the proof of Lemma 2.1.13.*

Proof. Note that $\alpha'' = M_0\alpha$, where $M_0 \in \Delta_0(\mathfrak{n})$ is an $(\mathfrak{a}, \mathfrak{a})$-matrix of level $\mathfrak{n}$, with $\mathfrak{a} = \langle \alpha' \rangle / \langle \alpha \rangle$. Now if $\alpha' = M\alpha$ with $M \in \Delta_0(\mathfrak{n})$, we have that $\alpha'' = M_0M^{-1}\alpha'$, where $M_0M^{-1} \in \Delta_0(\mathfrak{n})$. But since we have chosen α'' so that $[\alpha''] = [\alpha']$, by Proposition 1.2.24, $M_0M^{-1} \in \Gamma$. That is, $M_0M^{-1} \in \Gamma \cap \Delta_0(\mathfrak{n}) = \Gamma_0(\mathfrak{n})$, which proves that α' and α'' are $\Gamma_0(\mathfrak{n})$-equivalent. Conversely, suppose that $\alpha' = \gamma\alpha''$ with $\gamma \in \Gamma_0(\mathfrak{n})$. Then $\alpha' = \gamma\alpha'' = (\gamma M_0)\alpha$, and $\gamma M_0 \in \Delta_0(\mathfrak{n})$. □

Moreover, the number of $\Delta_0(\mathfrak{n})$-equivalence classes can also be easily deduced from the reduction to the $\Gamma_0(\mathfrak{n})$ case.

Corollary 2.1.15. *Write $h_K = \#\,\mathrm{Cl} = h_2 h_2'$, with $h_2 = \#\,\mathrm{Cl}[2]$ and $h_2' = \#\,\mathrm{Cl}\,/\,\mathrm{Cl}[2]$. The total number of $\Delta_0(\mathfrak{n})$-orbits of cusps is*

$$h_2' \sum_{\mathfrak{d}|\mathfrak{n}} \varphi_{\mathrm{u}}(\mathfrak{d} + \mathfrak{n}\mathfrak{d}^{-1}),$$

with

$$\varphi_{\mathrm{u}}(\mathfrak{m}) = \#((R/\mathfrak{m})^{\times}/U_{\mathfrak{m}}).$$

Proof. In the proof of Proposition 1.2.29 it was shown that, fixing an ideal class and choosing an ideal $\mathfrak{a}$ in this class, the number of $\Gamma_0(\mathfrak{n})$-orbits of cusps in $[\mathfrak{a}]$ is given by,

$$\sum_{\mathfrak{d}|\mathfrak{n}} \varphi_{\mathrm{u}}(\mathfrak{d} + \mathfrak{n}\mathfrak{d}^{-1}). \tag{2.1.1}$$

Let $[\mathfrak{a}_1], \ldots, [\mathfrak{a}_{h_2'}]$ be representatives of the cosets of $\mathrm{Cl}[2]$ in Cl. As we know, these ideal classes are in bijection with the Δ-orbits of cusps (Corollary 2.1.11). Now, in the previous result we have seen that to check $\Delta_0(\mathfrak{n})$-equivalence between cusps which are in the same Δ-class we need only to test for $\Gamma_0(\mathfrak{n})$-equivalence of cusps that have the same associated ideal class, say $[\mathfrak{a}_i]$. Hence each Δ-orbit of cusps splits in as many $\Delta_0(\mathfrak{n})$-orbits as $\Gamma_0(\mathfrak{n})$-orbits are in the representative class $[\mathfrak{a}_i]$, which is given by (2.1.1). □

About M-symbols

In J. Bygott's thesis [6] we find the following result:

Proposition 2.1.16. *If $\{S_i\}_{i\in I}$ is a set of (right) coset representatives for $\Gamma_0(\mathfrak{n})$ in Γ, then it is also a set of (right) coset representatives for $\Delta_0(\mathfrak{n})$ in Δ.*

Proof. See [6, Corollary 37] □

So in particular, the M-symbols of level $\mathfrak{n}$ (as we defined them in 1.2.4) not only provide a set of right coset representatives for $\Gamma_0(\mathfrak{n})$ in Γ, but a set of coset representatives for $\Delta_0(\mathfrak{n})$ in Δ as well.

2.2 The normaliser of $\Gamma_0(N)$ in $\mathrm{PSL}(2,\mathbb{R})$

We will denote the normaliser group of H in the group $G \supseteq H$ by $\mathcal{N}_G(H)$. For now we consider $G = \mathrm{SL}(2,\mathbb{R})$ and $H = \Gamma_0(N)$. The aim of this section is to obtain an explicit description of the elements in $\mathcal{N}_G(\Gamma_0(\mathfrak{n}))$. We begin with some useful properties of the congruence subgroup $\Gamma_0(N)$.

Lemma 2.2.1. *Define* $\varepsilon = \gcd(\{\, a-d : \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0(N) \,\})$. *Then* $\varepsilon = \gcd(N, 24)$.

Proof. This result is essentially proved in [23], where Newman finds the normaliser of $\Gamma_0(N)$ in the full modular group Γ. Here we give a straightforward proof.

First note that $\varepsilon | N$, since $\begin{pmatrix} N+1 & 1 \\ N & 1 \end{pmatrix} \in \Gamma_0(N)$. Then since $ad \equiv 1 \pmod{N}$, we have $ad \equiv 1 \pmod{\varepsilon}$. Now observe that by our definition of ε, $a \equiv d \pmod{\varepsilon}$. In particular we deduce that all units mod ε have order 2. Recall now that the unit group $(\mathbb{Z}/n\mathbb{Z})^\times$ is an elementary abelian 2-group if and only if n divides 24, and the Lemma follows. □

Lemma 2.2.2. *The* $\mathbb{Z}$*-span of* $\Gamma_0(N)$ *in* $\mathrm{Mat}_2(\mathbb{Z})$ *is*

$$\mathrm{M}_0(N) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{Mat}_2(\mathbb{Z}) : N|c \text{ and } \varepsilon|(a-d) \right\},$$

where $\varepsilon = \gcd(N, 24)$.

Proof. Let us denote the $\mathbb{Z}$-span of a set S by $\langle S \rangle_{\mathbb{Z}}$. It is clear that $\langle \Gamma_0(N) \rangle_{\mathbb{Z}} \subseteq \mathrm{M}_0(N)$. For the inclusion in the other direction, we observe that we can take as generators of $\mathrm{M}_0(N)$ the following elements:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} \varepsilon & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ N & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}. \tag{2.2.1}$$

Now we prove that these generators are all in the $\mathbb{Z}$-span of $\Gamma_0(N)$. This is trivial for the identity matrix and very easy for the last two:

$$\begin{aligned} \begin{pmatrix} 0 & 0 \\ N & 0 \end{pmatrix} &= \begin{pmatrix} 1 & 0 \\ N & 1 \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\ \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} &= \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \end{aligned}$$

The ε-generator requires more work. First note that we can write

$$\begin{pmatrix} a-d & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} - b\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} - \frac{c}{N}\begin{pmatrix} 0 & 0 \\ N & 0 \end{pmatrix} - d\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad (2.2.2)$$

for any $\begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0(N)$. Define

$$S_{a-d} = \langle\, \{a-d : \exists \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0(N)\,\}\, \rangle_{\mathbb{Z}}.$$

Since $\varepsilon = \gcd(N, 24)$, it follows that $\varepsilon \in S_{a-d}$ (recall Lemma 2.2.1). Now using (2.2.2) we can prove that:

$$S_{a-d} = \left\{ x : \begin{pmatrix} x & 0 \\ 0 & 0 \end{pmatrix} \in \langle \Gamma_0(N) \rangle_{\mathbb{Z}} \right\},$$

from which we deduce that the generator $\begin{pmatrix} \varepsilon & 0 \\ 0 & 0 \end{pmatrix}$ is in $\langle \Gamma_0(N) \rangle_{\mathbb{Z}}$. □

Remark 2.2.3. It is not difficult to check that $\mathrm{M}_0(N)$ is in fact a ring. Also, we now have that for any $M \in G = \mathrm{SL}(2, \mathbb{R})$,

$$M \in \mathcal{N}_G(\Gamma_0(N)) \iff M \in \mathcal{N}_G(\mathrm{M}_0(N)).$$

Thus we can characterise any element of $\mathcal{N}_G(\Gamma_0(N))$ as follows:

$$M \in \mathcal{N}_G(\Gamma_0(N)) \iff M\gamma M^{-1} \in \mathrm{M}_0(N),$$

for γ running through the generators given in (2.2.1).

The following Proposition describes the elements of the normaliser group. This result was first proved in [21]. We use a slightly different approach in our proof which we believe will be more amenable to a generalisation than the original proof in [21], and so we give all details of the necessary calculations.

Proposition 2.2.4. *Write $N = q\sigma^2$, with q square free, and let h be the largest divisor of 24 such that $h^2|N$. A matrix $M \in \mathrm{SL}(2, \mathbb{R})$ is in the normaliser $\mathcal{N}_G(\Gamma_0(N))$ if and only if*

$$M = \sqrt{e}\begin{pmatrix} aD & b/ehD \\ cN/ehD & dD \end{pmatrix}, \quad (2.2.3)$$

where $a, b, c, d, e, D, h \in \mathbb{Z}$, $e > 0$, $D|\frac{\sigma}{h}$ and $e|q$.

Proof. Let $M = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix}$ in $G = \mathrm{SL}(2,\mathbb{R})$. From our observations in Remark 2.2.3, we know that $M \in \mathcal{N}_G(\Gamma_0(N))$ if and only if the following hold:

$$\begin{aligned} M \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} M^{-1} &= \begin{pmatrix} -\alpha\gamma & \alpha^2 \\ -\gamma^2 & \alpha\gamma \end{pmatrix} \in \mathrm{M}_0(N), \\ M \begin{pmatrix} 0 & 0 \\ N & 0 \end{pmatrix} M^{-1} &= N \begin{pmatrix} \beta\delta & -\beta^2 \\ \delta^2 & \beta\delta \end{pmatrix} \in \mathrm{M}_0(N), \\ M \begin{pmatrix} \varepsilon & 0 \\ 0 & 0 \end{pmatrix} M^{-1} &= \varepsilon \begin{pmatrix} \alpha\delta & -\alpha\beta \\ \delta\gamma & -\beta\gamma \end{pmatrix} \in \mathrm{M}_0(N), \end{aligned}$$

where $\varepsilon = \gcd(N, 24)$. From these expressions we get the following sets of conditions on the coefficients of M:

(i) $\alpha\gamma,\ \alpha^2,\ \gamma^2 \in \mathbb{Z}$ and $N|\gamma^2$,

(ii) $N\beta^2,\ N\delta^2,\ N\beta\delta \in \mathbb{Z}$ and $N|N\delta^2$,

(iii) $\varepsilon\alpha\delta,\ \varepsilon\alpha\beta,\ \varepsilon\gamma\delta,\ \varepsilon\beta\gamma \in \mathbb{Z}$ and $N|\varepsilon\gamma\delta$.

Since $\det M = 1$, α and γ cannot both be zero. Suppose that $\alpha \neq 0$, then we can write:

$$\frac{\beta}{\alpha} = \frac{\varepsilon\alpha\beta}{\varepsilon\alpha^2}, \quad \frac{\delta}{\alpha} = \frac{\varepsilon\alpha\delta}{\varepsilon\alpha^2}, \quad \frac{\gamma}{\alpha} = \frac{\alpha\gamma}{\alpha^2},$$

and using conditions (i), (ii), (iii) we conclude that $\beta/\alpha, \delta/\alpha, \gamma/\alpha \in \mathbb{Q}$. In fact, since $\alpha^2 \in \mathbb{Z}$ by (i), we may write $\alpha = t\alpha_0$, with $t = 1$ if $\alpha \in \mathbb{Z}$ and otherwise choosing $t \in \mathbb{R}\backslash\mathbb{Q}$ such that $t^2 \in \mathbb{Z}$ and $\alpha_0 \in \mathbb{Q}$. Thus we have:

$$M = t \begin{pmatrix} \alpha_0 & \beta_0 \\ \gamma_0 & \delta_0 \end{pmatrix}, \tag{2.2.4}$$

where $\alpha_0, \beta_0, \gamma_0, \delta_0 \in \mathbb{Q}$. Now if $\alpha = 0$, then $\gamma \neq 0$ and we may use similar arguments to prove that:

$$M = t \begin{pmatrix} 0 & \beta_0 \\ \gamma_0 & \delta_0 \end{pmatrix},$$

where $\alpha_0, \beta_0, \gamma_0, \delta_0 \in \mathbb{Q}$ and as before, $t = 1$ if $\gamma \in \mathbb{Z}$ and otherwise $t \in \mathbb{R}\backslash\mathbb{Q}$ with $t^2 \in \mathbb{Z}$. Hence we can assume without loss of generality that (2.2.4) holds. Furthermore, since either $t = 1$ or $t \in \mathbb{R}\backslash\mathbb{Q}$ it is clear that $e = t^2 \in \mathbb{Z}$ is a square free integer, and from $e\alpha_0^2 = \alpha^2 \in \mathbb{Z}$ we deduce that in fact $\alpha_0 \in \mathbb{Z}$. The same argument applies to γ_0 and δ_0, so that $\gamma_0, \delta_0 \in \mathbb{Z}$.

Let $e_0 = \gcd(N, e)$, and write $e = e_0 e_1, N = N_0 e_0$ so that $\gcd(e_1, N_0) = 1$. Write $N_0 = q_0 \sigma^2$, with q_0 square free. We will now use the properties in (i), (ii) and (iii) to find out more about the coefficients $\alpha_0, \beta_0, \gamma_0, \delta_0$.

We start with the coefficient β_0. Say $\beta_0 = \beta_1/\beta_2$, with $\beta_1, \beta_2 \in \mathbb{Z}$ and coprime. Recall that $1 = \det(M) = e\alpha_0\delta_0 - e\gamma_0\beta_0$. It follows that $e\gamma_0\beta_0 = e\alpha_0\delta_0 - 1 \in \mathbb{Z}$ and $e\gamma_0\beta_0 = e\gamma_0\frac{\beta_1}{\beta_2}$ is coprime to e. Then e must divide β_2, so we can write $\beta_2 = e\beta_2'$ and

$$\beta_0 = \frac{\beta_1}{e\beta_2'}, \text{ with } \gcd(\beta_1, e\beta_2') = 1. \tag{2.2.5}$$

We now prove that $e|N$. From (ii), $N\beta^2 = Ne\beta_0^2 \in \mathbb{Z}.$, and by (2.2.5) this implies that $e(\beta_2')^2|N$. Hence $e|N$, and so $e_0 = \gcd(e, N) = e$, $e_1 = 1$ and $N = N_0 e$. We also have $\beta_2'|\sigma$, since $e(\beta_2')^2|N \Rightarrow (\beta_2')^2|N_0 = q_0\sigma^2$ and we have chosen q_0 square free. In particular we may rewrite (2.2.5) as follows:

$$\beta_0 = \frac{\beta_1}{e\beta_2'} = \frac{b'}{e\sigma}, \text{ with } b' \in \mathbb{Z}, \tag{2.2.6}$$

where we have taken $b' = \beta_1\frac{\sigma}{\beta_2'}$.

From (i) we know that $N|\gamma^2 = e\gamma_0^2$. In particular $N_0|\gamma_0^2$, and thus $q_0\sigma|\gamma_0$. We now have

$$\gamma_0 = c'\sigma q_0, \text{ for some } c' \in \mathbb{Z}. \tag{2.2.7}$$

Next we are going to prove that the product $q = eq_0$ is square free, which means that we have a decomposition $N = q\sigma^2$ like in the statement of the Proposition, with $e|q$. Using (2.2.5), (2.2.6) and (2.2.7) we obtain:

$$1 = \det(M) = e\alpha_0\delta_0 - e\gamma_0\beta_0 = e\alpha_0\delta_0 - c'q_0b'. \tag{2.2.8}$$

From this we see that q_0 and e are coprime and thus $q = eq_0$ is square free.

Let h be the largest divisor of 24 such that $h^2|N$. By (iii), $\varepsilon\alpha\beta \in \mathbb{Z}$ and $N|\varepsilon\gamma\delta$. Then

$$\varepsilon\alpha\beta \in \mathbb{Z} \;\Rightarrow\; \varepsilon\alpha_0\frac{b'}{\sigma} \in \mathbb{Z} \;\Rightarrow\; \sigma|\varepsilon\alpha_0 b' \;\Rightarrow\; \frac{\sigma}{h}|\alpha_0 b',$$

and

$$N|\varepsilon\delta\gamma = \varepsilon e\delta_0 c'\sigma q_0 \;\Rightarrow\; \sigma|\varepsilon\delta_0 c' \;\Rightarrow\; \frac{\sigma}{h}|\delta_0 c',$$

since $\varepsilon = \gcd(N, 24)$. Now take $D = \gcd(\sigma/h, \alpha_0)$. From (2.2.8) it follows that α_0 is coprime to $b'c'$. Hence D must divide δ_0 and α_0. Also, $\sigma/(hD)$ divides both b'

and c'. Thus we can write:

$$\alpha_0 = aD \text{ and } \delta_0 = dD, \text{ with } a, d \in \mathbb{Z}.$$

Finally taking $b, c \in \mathbb{Z}$ such that $b' = \frac{b\sigma}{hD}$ and $c' = \frac{c\sigma}{hD}$ we see that

$$\beta_0 = \frac{b'}{e\sigma} = \frac{b}{ehD} \text{ and } \gamma_0 = cq_0\sigma\frac{\sigma}{hD} = c\frac{N}{ehD}.$$

□

Remark 2.2.5. Some more work is done in the paper by Lehner and Newman [21] concerning the structure of the normaliser group in the special case when $h = 1$. However there is a mistake and Theorem 3 as stated in the paper is wrong. This was later corrected in [2], where some other mistakes were made. But we look at the structure of the normaliser in the next section.

2.2.1 The structure of the normaliser group

In their classical paper [2] Atkin and Lehner gave for the first time a full description of the structure of the normaliser group $\mathcal{N}_G(\Gamma_0(N))$. Their result follows the work by Lehner and Newman in [21]. It is stated without a proof at the end of the paper ([2, Theorem 8]), and it gives a rather complicated description of the normaliser. A much clearer exposition on the subject can be found in [1], where Akbas and Singerman use an alternative description for the elements of the normaliser first introduced by Conway and Norton [8]. Akbas and Singerman's paper also corrects some of the claims in [2] which were wrong.

Remark 2.2.6. Another correct description of the structure of the normalizer that follows the original formulation of Atkin and Lehner [2] is given in a paper by F. Bars [3].

We now give a brief description of the structure of the normaliser of $\Gamma_0(N)$, which served as inspiration for our work in the next section for the general number field case. We follow the work of Akbas and Singerman [1].

Let $\mathrm{PSL}(2, \mathbb{R})$ be the group of all Möbius transformations with real coefficients and determinant one. We will now consider the normaliser of $\Gamma_0(N)$ in $\mathrm{PSL}(2, \mathbb{R})$, i.e. $\mathcal{N}_G(\Gamma_0(N))$ with $G = \mathrm{PSL}(2, \mathbb{R})$. For simplicity, from now on we will denote this normaliser group by $\mathcal{N}(N)$.

The elements of $\mathcal{N}(N)$ are given by the transformations corresponding to the

matrices

$$\begin{pmatrix} aE & b/h \\ cN/h & dE \end{pmatrix}, \tag{2.2.9}$$

where all symbols represent integers, h is the largest divisor of 24 such that $h^2|n$, and the determinant of the matrix is E, with $E > 0$ an exact divisor of N (that is, $\gcd(E, N/E) = 1$). This description of the normaliser (first given by Conway and Norton [8]) can be obtained from Proposition 2.2.4. We need only to multiply the matrix in (2.2.3) by a convenient scalar (namely $\sqrt{e}D$ with e, D as given in Proposition 2.2.4).

The matrices of the form (2.2.9) form a group up to scalar multiplication. Now we define a special subset of the normaliser group:

Definition 2.2.7. We will denote by $\mathcal{N}_h(N)$ the set of transformations of the form (2.2.9) with determinant $E = 1$.

$\mathcal{N}_h(N)$ is a conjugate of $\Gamma_0(N/h^2)$ by the matrix $\begin{pmatrix} h & 0 \\ 0 & 1 \end{pmatrix}$. In fact, it is not difficult to see that $\mathcal{N}_h(N)$ is a normal subgroup of $\mathcal{N}(N)$. Hence we have $\Gamma_0(N) \lhd \mathcal{N}_h(N) \lhd \mathcal{N}(N)$. In order to understand the elements of the normaliser which are not contained in $\mathcal{N}_h(N)$ is necessary to introduce the following special transformations.

Definition 2.2.8. Let $E > 0$ be any exact divisor of N. The *Atkin-Lehner transformation* W_E is represented by the matrix

$$W_E = \begin{pmatrix} aE & b \\ cN & dE \end{pmatrix}, \tag{2.2.10}$$

with $\det(W_E) = E$. Note that $W_E \in \mathcal{N}(N)$.

All transformations of the form (2.2.10) with a given E belong to the same $\Gamma_0(N)$-coset in $\mathcal{N}(N)$. Hence we can use the notation W_E to represent any of them. The special case $E = N$, where $W_N = \begin{pmatrix} 0 & 1 \\ -N & 0 \end{pmatrix}$, is called *Fricke transformation.*

Furthermore, up to scalar multiplication, Atkin-Lehner transformations form a group, which we will denote $\mathcal{N}_W(N)$. The quotient group $\mathcal{N}_W(N)/\Gamma_0(N)$ is an elementary abelian 2-group of order 2^r, where r is the number of prime divisors of N. Consequently, for all Atkin-Lehner transformations W_E, $W_E^2 \in \Gamma_0(N)$ (modulo scalars) .

Proposition 2.2.9. *Every element M of $\mathcal{N}(N)$ can be written as a product WT, where $W \in \mathcal{N}_W(N)$ and $T \in \mathcal{N}_h(N)$.*

Proof. [1, Proposition 3]. □

The structure of the quotient group $\mathcal{N}(N)/\Gamma_0(N)$ is thoroughly described by Akbas and Singerman [1]. Here we note only that since

$$W(N) = \mathcal{N}_W(N)/\Gamma_0(N) \cong C_2^r,$$

all the elements of $W(N) - \{I\}$ commute and have order two. They are called *Atkin-Lehner involutions*.

2.2.2 The normaliser of $\Gamma_0^{\pm}(N)$

The above discussion considers $\Gamma_0(N)$ as a subgroup of the modular group $\mathrm{SL}(2,\mathbb{Z})$. However in our work for the number field case we use $\mathrm{GL}(2)$ instead of $\mathrm{SL}(2)$. In this section we look at the situation over $\mathbb{Q}$ when working in $\mathrm{GL}(2,\mathbb{Z})$. The results for the normaliser of $\Gamma_0(N)$ can be easily adapted for the group $\Gamma_0^{\pm}(N)$, which we define by

$$\begin{aligned}\Gamma_0^{\pm}(N) &= \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{GL}(2,\mathbb{Z}) : c \equiv 0 \pmod{N} \right\} \\ &= \Gamma_0(N) \cup \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \Gamma_0(N).\end{aligned}$$

Lemma 2.2.10. *Define* $\varepsilon = \gcd(\{ a-d : \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0^{\pm}(N) \})$. *Then* $\varepsilon = \gcd(N,2)$.

Proof. As in the proof of Lemma 2.2.1, we have $\varepsilon | N$. On the other hand, ε also divides 2 since $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \in \Gamma_0^{\pm}(N)$, and then the Lemma follows. □

Lemma 2.2.11. *The $\mathbb{Z}$-span of $\Gamma_0^{\pm}(N)$ in $\mathrm{Mat}_2(\mathbb{Z})$ is*

$$\mathrm{M}_0^{\pm}(N) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{Mat}_2(\mathbb{Z}) : N|c \text{ and } \varepsilon|(a-d) \right\},$$

where $\varepsilon = \gcd(N,2)$.

Proof. Analogous to the proof of Lemma 2.2.2. □

Now define the group $\mathrm{SL}^{\pm}(2,\mathbb{R}) = \{M \in \mathrm{Mat}_2(\mathbb{R}) : \det(M) = \pm 1\}$. Clearly $\mathrm{GL}(2,\mathbb{R}) = \mathbb{R}^{\times} \cdot \mathrm{SL}^{\pm}(2,\mathbb{R})$. Hence the normaliser of $\Gamma_0^{\pm}(N)$ in $\mathrm{SL}^{\pm}(2,\mathbb{R})$ determines, up to product by scalar matrices, the normaliser of $\Gamma_0^{\pm}(N)$ in the bigger group $\mathrm{GL}(2,\mathbb{R})$.

Proposition 2.2.12. *Write $N = q\sigma^2$, with q square free, and define*

$$h = \begin{cases} 1 & \text{if } 4 \nmid N, \\ 2 & \text{if } 4|N. \end{cases}$$

Then a matrix $M \in G = \mathrm{SL}^{\pm}(2,\mathbb{R})$ is in the normaliser $\mathcal{N}_G(\Gamma_0^{\pm}(N))$ if and only if

$$M = \sqrt{e} \begin{pmatrix} aD & b/ehD \\ cN/ehD & dD \end{pmatrix}, \tag{2.2.11}$$

where

$$a, b, c, d, e, D \in \mathbb{Z}, \quad D|\frac{\sigma}{h}, \quad e|q.$$

Proof. We can repeat the proof of Proposition 2.2.4. We need only to substitute the definitions and properties in Lemmas 2.2.1 and 2.2.2 by the corresponding ones given in Lemmas 2.2.10 and 2.2.11. □

Following the ideas in §2.2.1, we now consider the normaliser in the group $\mathrm{PGL}(2,\mathbb{R})$. Clearly the elements of $\mathcal{N}_G(\Gamma_0^{\pm}(N))$, for $G = \mathrm{PGL}(2,\mathbb{R})$, are given by the transformations corresponding to the matrices

$$\begin{pmatrix} aE & b/h \\ cN/h & dE \end{pmatrix}, \tag{2.2.12}$$

where all symbols represent integers, h is 1 or 2 (as in Proposition 2.2.12), and the determinant of the matrix is E, with E an exact divisor of N. In particular, our previous discussion in §2.2.1 about the structure of $\mathcal{N}(N) = \mathcal{N}_G(\Gamma_0(N))$, with $G = \mathrm{PSL}(2,\mathbb{R})$, will apply to the current situation with only minor changes (such as the definition of the variable h).

2.3 The normaliser of $\Gamma_0(\mathfrak{n})$

In this section we generalise some of the results in §2.2 for the general number field case. However, we have not obtained a complete description of the normaliser of $\Gamma_0(\mathfrak{n})$ such as the one given in Proposition 2.2.4 for the classical case. Recall the definition of the congruence subgroup $\Gamma_0(\mathfrak{n})$:

$$\Gamma_0(\mathfrak{n}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma \,|\, c \in \mathfrak{n} \right\}.$$

We now prove analogues of Lemmas 2.2.1 and 2.2.2 for the number field case.

Lemma 2.3.1. *Define* $\mathcal{E} = \langle \{ a - d : \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0(\mathfrak{n}) \} \rangle_R$, *an integral ideal. Then* $\mathcal{E} = \mathcal{E}_u + \mathfrak{n}$, *where*

$$\mathcal{E}_u = \langle \{ u - 1 : u \in R^\times \} \rangle_R .$$

Proof. We follow the ideas in the proof of the equivalent result over $\mathbb{Q}$, Lemma 2.2.10. Again we have $\mathfrak{n} \subseteq \mathcal{E}$. On the other hand, $\begin{pmatrix} u & 0 \\ 0 & 1 \end{pmatrix} \in \Gamma_0(\mathfrak{n})$ for all $u \in R^\times$, so that clearly $\mathcal{E}_u \subseteq \mathcal{E}$, and then the Lemma follows. □

The ideal $\mathcal{E}_u$ can be computed considering only a set of generators for the units in the multiplicative group $R^\times$. It is also clear that $\mathcal{E}_u | \langle 2 \rangle$ (or equivalently, $\mathcal{E}_u \supseteq \langle 2 \rangle$), since $-1 \in R^\times$ always. For instance, for $K = \mathbb{Q}(i)$, with $i = \sqrt{-1}$, we have $R^\times = \langle i \rangle$ and then $\mathcal{E}_u = \langle 1 + i \rangle_R$.

Remark 2.3.2. If working in SL(2) instead of GL(2), it is not difficult either to generalise the corresponding description in Lemma 2.2.1. Indeed, we have

$$\mathcal{E} = \langle \{ a - d : \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma_0(\mathfrak{n}) \} \rangle_R = \mathcal{E}_{24} + \mathfrak{n},$$

where $\mathcal{E}_{24}$ is the ideal in R characterised by the following property: $(R/\mathfrak{a})^\times$ is an elementary abelian 2-group if and only if $\mathfrak{a} | \mathcal{E}_{24}$. From [7, Proposition 4.2.12] we obtain a complete description of this ideal:

$$\mathcal{E}_{24} = \prod_{\mathrm{N}(\mathfrak{p})=2} \mathfrak{p}^e \prod_{\mathrm{N}(\mathfrak{q})=3} \mathfrak{q},$$

where $e = 3$ if $\mathfrak{p}$ is unramified, and $e = 2$ otherwise.

Lemma 2.3.3. *The R-span of $\Gamma_0(\mathfrak{n})$ in $\mathrm{Mat}_2(R)$ is*

$$\mathrm{M}_0(\mathfrak{n}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \mathrm{Mat}_2(R) : c \in \mathfrak{n} \text{ and } a - d \in \mathcal{E} \right\},$$

with $\mathcal{E} = \mathcal{E}_u + \mathfrak{n}$.

Proof. It is clear that $\langle \Gamma_0(\mathfrak{n}) \rangle_R \subseteq \mathrm{M}_0(\mathfrak{n})$. For the converse, observe that the following matrices are a set of generators for $\mathrm{M}_0(\mathfrak{n})$ over R:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} e_i & 0 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ n_i & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \tag{2.3.1}$$

for $i \in \{1, 2\}$, and with $\mathcal{E} = \langle e_1, e_2 \rangle_R$, $\mathfrak{n} = \langle n_1, n_2 \rangle_R$. Using the strategies of Lemma 2.2.2 one can check that all these matrices belong to the R-span of $\Gamma_0(\mathfrak{n})$, and this proves the Lemma. □

Using techniques similar to the ones in Proposition 2.2.4 and Lemmas 2.3.1 and 2.3.3 above, it may be possible to obtain a general description of the elements of the normaliser group of $\Gamma_0(\mathfrak{n})$ in $\mathrm{PGL}(2, \mathbb{C})$ (or $\mathrm{PSL}(2, \mathbb{C})$). We will not pursue this here (see Remark 2.3.5 below), though we easily obtain some partial results.

Let L be any extension of the field K and $M \in G = \mathrm{GL}(2, L)$ a matrix which normalises $\Gamma_0(\mathfrak{n})$. Write

$$M = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix}.$$

We now follow the first steps in the proof of Proposition 2.2.4. We can assume $\alpha \neq 0$ (otherwise $\gamma \neq 0$ and a similar result follows, exactly as in Proposition 2.2.4). Then M will satisfy:

$$M = t \begin{pmatrix} \alpha_0 & \beta_0 \\ \gamma_0 & \delta_0 \end{pmatrix}, \tag{2.3.2}$$

where $\alpha_0, \beta_0, \gamma_0, \delta_0 \in K$ and we take $t = 1$ if $\alpha^2 \in K$ and otherwise $t \in L \backslash \{0\}$ such that $t^2 \in K$. That is, a matrix in the normaliser is in $\mathrm{GL}(2, K)$ up to scalar factors. Furthermore, the rest of the arguments in the proof of Proposition 2.2.4 hold if the ring of integers of K is a unique factorization domain. This leads to the following partial result.

Proposition 2.3.4. *Let K be a number field with class number $h_K = 1$, R its ring of integers and L an extension of K. Write $\mathfrak{n} = \mathfrak{q}\mathfrak{s}^2$, with $\mathfrak{q}$ a square free integral ideal, and let $\mathcal{E}_u$ be as in* Lemma 2.3.1. *Denote by $\mathfrak{h}$ the biggest divisor of the ideal $\mathcal{E}_u$ whose square divides $\mathfrak{n}$. Then a matrix $M \in \mathrm{GL}(2, L)$ normalises $\Gamma_0(\mathfrak{n})$ if and only if, up to scalar factors, M is of the form:*

$$\begin{pmatrix} aD & b/ehD \\ cN/ehD & dD \end{pmatrix},$$

where $a, b, c, d, e, D, h \in R$ and

$$\langle D\rangle | \mathfrak{s}\mathfrak{h}^{-1},\ \langle e\rangle | \mathfrak{q},\ \langle h\rangle | \mathfrak{h}.$$

Proof. Since $h_K = 1$, R is a principal ideal domain and in particular a unique factorization domain. Hence we can repeat the steps in the proof of Proposition 2.2.4. □

Back in the general case, from our comments above (equation (2.3.2)) it seems reasonable to restrict our interest to the normaliser of $\Gamma_0(\mathfrak{n})$ in $\mathrm{GL}(2, K)$. When $\mathfrak{n} = R$, i.e. $\Gamma_0(\mathfrak{n}) = \Gamma$, we already have a complete answer. In §2.1 we defined $\Delta = \mathcal{N}_G(\Gamma)$, with $G = \mathrm{GL}(2, K)$, and we saw that Δ can be described as the set of $(\mathfrak{a}, \mathfrak{a})$-matrices, for all ideals $\mathfrak{a}$ such that its square is principal (Theorem 2.1.1 and Lemma 2.1.2). In general we have Atkin-Lehner type transformations, which we discuss in the next section. The elements of Δ are a particular case of these more general transformations.

Looking at the models given by the rational case and Proposition 2.3.4, we would expect to find elements in the normaliser group that do not come from Atkin-Lehner type transformations. These elements will be analogues to the transformations in $\mathcal{N}_h(N)$ of Definition 2.2.7 and should satisfy some conditions involving the special ideal $\mathcal{E}$ defined in Lemma 2.3.1. Although more work is needed to find a general description of this type of elements, we now give an easy example. Using the techniques in Proposition 2.2.4 (namely checking that our element normalises the generators given in (2.3.1)) one can prove that, for $\alpha \in K$,

$$\begin{pmatrix} 1 & \alpha \\ 0 & 1 \end{pmatrix} \in \mathcal{N}_{\mathrm{GL}(2,K)}(\Gamma_0(\mathfrak{n})) \Longleftrightarrow \mathfrak{d}(\alpha) | \mathfrak{h},$$

where $\mathfrak{d}(\alpha)$ denotes the denominator ideal of α (recall Definition 1.2.25) and the ideal $\mathfrak{h}$ is the biggest divisor of $\mathcal{E}_u$ whose square divides the level (exactly as in

Proposition 2.3.4). This also suggests that, as in the rational case, when $\mathfrak{h} = R$ (e.g. if the ideal $\mathcal{E}_u$ is coprime to the level) all elements of the normaliser come from Atkin-Lehner transformations.

Remark 2.3.5. One motivation for considering elements which normalise $\Gamma_0(\mathfrak{n})$ lies in the fact that such an element will induce an operator on any space on which Γ acts, preserving $\Gamma_0(\mathfrak{n})$-invariant subspaces. In particular, we can use these operators to obtain more information about spaces of modular forms. In previous computations for imaginary quadratic fields of small class number ([6] and [22]) enough information was obtained taking into account only transformations of the type described in §2.3.1 below.

In any case, if more information has to be obtained from operators coming from normaliser elements, it is possible that it should come from a more general normaliser group. This would be the normaliser for "twisted" versions of $\Gamma_0(\mathfrak{n})$, which are defined by:

$$\Gamma_0^{\mathfrak{a}}(\mathfrak{n}) = \left\{ \begin{pmatrix} a & b \\ c & d \end{pmatrix} : \ a,\, d \in R,\, b \in \mathfrak{a}^{-1},\, c \in \mathfrak{a}\mathfrak{n} \ \text{ and } \ ad - bc \in R^{\times} \right\},$$

for any integral ideal $\mathfrak{a}$. These "twisted" versions of the congruence subgroup $\Gamma_0(\mathfrak{n})$ play an important role in the theory of modular forms and Hecke operators over number fields.

2.3.1 Atkin-Lehner type transformations

In this section we finish our discussion of normaliser elements with a description of Atkin-Lehner type transformations. These transformations generalise the elements of the normaliser group Δ and the matrices introduced in [22, §5.3], which were defined for fields with odd class number.

Fix a level $\mathfrak{n}$. Let $\mathfrak{q}$ be an exact divisor of $\mathfrak{n}$, that is, a divisor of $\mathfrak{n}$ such that $\mathfrak{q}$ and $\mathfrak{n}\mathfrak{q}^{-1}$ are coprime; we will write $\mathfrak{q}||\mathfrak{n}$. Define $\mathfrak{q}' = \mathfrak{n}\mathfrak{q}^{-1}$. Whenever the ideal class $[\mathfrak{q}]$ is a square we can make the following construction.

Definition 2.3.6. Let $\mathfrak{m}$ be an ideal coprime to $\mathfrak{q}'$ such that $\mathfrak{q}\mathfrak{m}^2$ is principal. A $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix of level $\mathfrak{n}$ is a matrix $M = \begin{pmatrix} x & y \\ z & w \end{pmatrix}$ such that

$$M \in \begin{pmatrix} \mathfrak{m}\mathfrak{q} & \mathfrak{m} \\ \mathfrak{m}\mathfrak{n} & \mathfrak{m}\mathfrak{q} \end{pmatrix} \quad \textit{with} \quad \langle \det M \rangle = \mathfrak{q}\mathfrak{m}^2.$$

If we replace the ideal $\mathfrak{m}$ in the definition by an alternative ideal $\mathfrak{m}'$ in the same class, it is clear that $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices and $W_{\mathfrak{q}}^{\mathfrak{m}'}$-matrices differ only by a scalar factor. From now on we will then identify two such matrices, so that the set of $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices is in fact associated with a pair $(\mathfrak{q}, [\mathfrak{m}])$ such that $[\mathfrak{q}\mathfrak{m}^2]$ is trivial. Note that the number of such pairs is finite, so that modulo scalars we have a finite number of $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices.

Remark 2.3.7. For an analogue of the classical Fricke involution we need to use a $W_{\mathfrak{n}}^{\mathfrak{m}}$-matrix of level $\mathfrak{n}$, but such a matrix only exists if the ideal class of the level $\mathfrak{n}$ is a square.

We observe that for $\mathfrak{q} = \mathfrak{m} = R$, we obtain an element of $\Gamma_0(\mathfrak{n})$. If $\mathfrak{n} = R$, then $\mathfrak{q} = R$ and $\mathfrak{m}^2$ is principal, and we have an $(\mathfrak{m}, \mathfrak{m})$-matrix. In particular, the $W_R^{\mathfrak{m}}$-matrices of level R generate the group Δ defined in §2.1, which was the normaliser of Γ in $\mathrm{GL}(2, K)$ modulo scalars. If we only impose the condition $\mathfrak{q} = R$, we obtain $(\mathfrak{m}, \mathfrak{m})$-matrices of level $\mathfrak{n}$. That is, for level $\mathfrak{n}$ the $W_R^{\mathfrak{m}}$-matrices generate $\Delta_0(\mathfrak{n})$, a subgroup of the normaliser Δ.

In general, a $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix can be described as an $(\mathfrak{m}\mathfrak{q}, \mathfrak{m})$-matrix of level $\mathfrak{n}$: if $M \in W_{\mathfrak{q}}^{\mathfrak{m}}$, clearly $\langle \mathfrak{m}\mathfrak{q}, \mathfrak{m}\mathfrak{n} \rangle = \langle \mathfrak{m}\mathfrak{q} \rangle$, $\langle \mathfrak{m}, \mathfrak{m}\mathfrak{q} \rangle = \langle \mathfrak{m} \rangle$ and $\mathfrak{q}\mathfrak{m}^2 = \langle \det M \rangle$. It is also clear from Definition 2.3.6 that, up to multiplication by scalars, the inverse of a $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix is a $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix as well. We will now discuss the existence and uniqueness of $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices.

Proposition 2.3.8. (Existence of $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices) *There exist $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices for every $\mathfrak{q}||\mathfrak{n}$ and $\mathfrak{m}$ such that $\mathfrak{m}^2\mathfrak{q}$ is principal.*

Proof. Take an ideal $\mathfrak{a}$ in the inverse class to $\mathfrak{m}\mathfrak{n}$ and coprime to $\mathfrak{m}\mathfrak{n}$. Then let $z \in R$ be such that $\langle z \rangle = \mathfrak{a}\mathfrak{m}\mathfrak{n}$, so that in particular $z \in \mathfrak{m}\mathfrak{n}$.

Now let $\mathfrak{b}$ be an ideal in the inverse class to $\mathfrak{m}\mathfrak{q}$ and coprime to $\mathfrak{a}\mathfrak{q}'$, and take $x \in R$ so that $\langle x \rangle = \mathfrak{b}\mathfrak{m}\mathfrak{q}$, and so we have $x \in \mathfrak{m}\mathfrak{q}$.

Let $\mathfrak{q}\mathfrak{m}^2 = \langle g \rangle$. We note that

$$\langle gx, z \rangle = \mathfrak{m}^2\mathfrak{q}\mathfrak{b}\mathfrak{m}\mathfrak{q} + \mathfrak{a}\mathfrak{m}\mathfrak{n} = \mathfrak{m}\mathfrak{q}(\mathfrak{b}\mathfrak{m}^2\mathfrak{q} + \mathfrak{a}\mathfrak{q}') = \mathfrak{m}\mathfrak{q},$$

where the last equality holds since for this choice of $\mathfrak{a}$ and $\mathfrak{b}$, $\mathfrak{b}\mathfrak{m}^2\mathfrak{q}$ is coprime to $\mathfrak{a}\mathfrak{q}'$. It follows that $g \in \mathfrak{m}^2\mathfrak{q} = \mathfrak{m}\langle gx, z \rangle$, and hence there exist $y, w \in \mathfrak{m}$ such that $g = gxw - zy$. Now $\begin{pmatrix} x & y \\ z & gw \end{pmatrix}$ is a $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix of level $\mathfrak{n}$. □

Proposition 2.3.9 (Products of $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices)**.** *For $i \in \{1,2\}$, let M_i be a $W_{\mathfrak{q}_i}^{\mathfrak{m}_i}$-matrix of level $\mathfrak{n}$, with $\mathfrak{q}_i || \mathfrak{n}$ and $\mathfrak{q}_i \mathfrak{m}_i{}^2$ a principal ideal. Then, up to scalars, $M_3 = M_1 M_2$ is a $W_{q_3}^{\mathfrak{m}_3}$-matrix, with $\mathfrak{a} = \mathfrak{q}_1 + \mathfrak{q}_2$, $\mathfrak{m}_3 = \mathfrak{a}\mathfrak{m}_1\mathfrak{m}_2$ and $\mathfrak{q}_3 = \mathfrak{q}_1\mathfrak{q}_2\mathfrak{a}^{-2}$.*

Proof. Write $\mathfrak{q}_1 = \mathfrak{a}\mathfrak{q}_1'$ and $\mathfrak{q}_2 = \mathfrak{a}\mathfrak{q}_2'$, with $\mathfrak{a} = \mathfrak{q}_1 + \mathfrak{q}_2$. Then $\mathfrak{q}_3 = \mathfrak{q}_1'\mathfrak{q}_2' = \mathfrak{q}_1\mathfrak{q}_2\mathfrak{a}^{-2}$ is another divisor of $\mathfrak{n}$, and it is coprime to $\mathfrak{n}\mathfrak{q}_3^{-1}$ (since $\mathfrak{q}_i'$ is coprime to $\mathfrak{n}(\mathfrak{q}_i')^{-1}$, for $i \in \{1,2\}$). We also note that for $\mathfrak{m}_3 = \mathfrak{a}\mathfrak{m}_1\mathfrak{m}_2$, $\mathfrak{q}_3\mathfrak{m}_3^2$ is a principal ideal, since

$$\mathfrak{q}_3\mathfrak{m}_3 = \mathfrak{q}_3(\mathfrak{a}\mathfrak{m}_1\mathfrak{m}_2)^2 = \mathfrak{q}_1\mathfrak{q}_2\mathfrak{a}^{-2}(\mathfrak{a}\mathfrak{m}_1\mathfrak{m}_2)^2 = (\mathfrak{q}_1\mathfrak{m}_1^2)(\mathfrak{q}_2\mathfrak{m}_2^2).$$

Also, M_3 has the right determinant to be a $W_{\mathfrak{q}_3}^{\mathfrak{m}_3}$-matrix of level $\mathfrak{n}$:

$$\langle \det M_3 \rangle = \langle \det M_1 \rangle \langle \det M_2 \rangle = (\mathfrak{q}_1\mathfrak{m}_1^2)(\mathfrak{q}_2\mathfrak{m}_2^2) = \mathfrak{q}_3\mathfrak{m}_3^2.$$

Finally, observe that

$$\begin{aligned} M_3 &\in \begin{pmatrix} \mathfrak{m}_1\mathfrak{q}_1 & \mathfrak{m}_1 \\ \mathfrak{m}_1\mathfrak{n} & \mathfrak{m}_1\mathfrak{q}_1 \end{pmatrix} \begin{pmatrix} \mathfrak{m}_2\mathfrak{q}_2 & \mathfrak{m}_2 \\ \mathfrak{m}_2\mathfrak{n} & \mathfrak{m}_2\mathfrak{q}_2 \end{pmatrix} \\ &\subseteq \begin{pmatrix} \mathfrak{m}_1\mathfrak{m}_2(\mathfrak{q}_1\mathfrak{q}_2 + \mathfrak{n}) & \mathfrak{m}_1\mathfrak{m}_2(\mathfrak{q}_1 + \mathfrak{q}_2) \\ \mathfrak{m}_1\mathfrak{m}_2\mathfrak{n}(\mathfrak{q}_1 + \mathfrak{q}_2) & \mathfrak{m}_1\mathfrak{m}_2(\mathfrak{q}_1\mathfrak{q}_2 + \mathfrak{n}) \end{pmatrix} = \begin{pmatrix} \mathfrak{m}_3\mathfrak{q}_3 & \mathfrak{m}_3 \\ \mathfrak{m}_3\mathfrak{n} & \mathfrak{m}_3\mathfrak{q}_3 \end{pmatrix}, \end{aligned}$$

since $\mathfrak{q}_1 + \mathfrak{q}_2 = \mathfrak{a}$ and $\mathfrak{q}_1\mathfrak{q}_2 + \mathfrak{n} = \mathfrak{a}^2\mathfrak{q}_3 + \mathfrak{n} = \mathfrak{a}\mathfrak{q}_3(\mathfrak{a} + \mathfrak{n}\mathfrak{q}_3^{-1}\mathfrak{a}^{-1}) = \mathfrak{a}\mathfrak{q}_3$. □

Proposition 2.3.10 (Uniqueness of $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices)**.** *Let M_1, M_2 and M be $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices of level $\mathfrak{n}$ (with the same $\mathfrak{q}$ and $\mathfrak{m}$). Then:*

(i) *$M_1 M_2 \in \Gamma_0(\mathfrak{n})$ (up to a scalar factor).*

(ii) *$M_1 M_2^{-1} \in \Gamma_0(\mathfrak{n})$ and $M_1^{-1} M_2 \in \Gamma_0(\mathfrak{n})$.*

(iii) *The set of all $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices of level $\mathfrak{n}$ equals the left coset $M\Gamma_0(\mathfrak{n})$ and also the right coset $\Gamma_0(\mathfrak{n})M$.*

(iv) *If $\mathfrak{m}_1 \neq \mathfrak{m}$ is another ideal such that $\mathfrak{q}\mathfrak{m}_1^2$ is principal, say $\mathfrak{m}_1 = \mathfrak{m}\mathfrak{b}$ with $\mathfrak{b}^2$ principal, then the set of all $W_{\mathfrak{q}}^{\mathfrak{m}_1}$-matrices of level $\mathfrak{n}$ is*

$$\{MB \,|\, B \text{ is a } (\mathfrak{b},\mathfrak{b})\text{-matrix of level } \mathfrak{n}\}.$$

Proof. Part (i) can be deduced from the previous result. Write $M_3 = M_1 M_2$, then the ideals $\mathfrak{m}_3$, $\mathfrak{q}_3$ and $\mathfrak{a}$ in Proposition 2.3.9 for $\mathfrak{q}_1 = \mathfrak{q}_2 = \mathfrak{q}$ are given by $\mathfrak{a} = \mathfrak{q}$,

$\mathfrak{m}_3 = \mathfrak{q}\mathfrak{m}^2$ and $\mathfrak{q}_3 = R$. In particular, if we take $\langle g \rangle = \mathfrak{q}\mathfrak{m}^2$ we have

$$M_3 \in \begin{pmatrix} \mathfrak{m}_3\mathfrak{q}_3 & \mathfrak{m}_3 \\ \mathfrak{m}_3\mathfrak{n} & \mathfrak{m}_3\mathfrak{q}_3 \end{pmatrix} = \begin{pmatrix} \langle g \rangle & \langle g \rangle \\ \langle g \rangle \mathfrak{n} & \langle g \rangle \end{pmatrix} = \begin{pmatrix} g & 0 \\ 0 & g \end{pmatrix} \begin{pmatrix} R & R \\ \mathfrak{n} & R \end{pmatrix},$$

so clearly, up to scalars $M_1 M_2 \in \Gamma_0(\mathfrak{n})$.

For (ii), recall that for $i \in \{1, 2\}$, M_i^{-1} is also (up to scalars) a $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix of level $\mathfrak{n}$. Then (ii) follows from applying (i) to M_i and M_j^{-1}, for $i, j \in \{1, 2\}$.

Again using Proposition 2.3.9, it is clear that both cosets $M\Gamma_0(\mathfrak{n})$ and $\Gamma_0(\mathfrak{n})M$ consist of $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices of level $\mathfrak{n}$ (we may think of $\Gamma_0(\mathfrak{n})$ as the set of W_R^R-matrices of level $\mathfrak{n}$). Using (ii) we deduce that every $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix is in both cosets, and (iii) follows.

Part (iv) requires a similar calculation. Let $\mathfrak{m}_1$ be such that $\mathfrak{q}\mathfrak{m}_1^2$ is principal. Then there exists an ideal $\mathfrak{b}$ such that $\mathfrak{m}_1 = \mathfrak{m}\mathfrak{b}$ and $\mathfrak{b}^2$ is principal. Recall that $(\mathfrak{b}, \mathfrak{b})$-matrices of level $\mathfrak{n}$ are in fact $W_R^{\mathfrak{b}}$-matrices of level $\mathfrak{n}$. Let B be any $(\mathfrak{b}, \mathfrak{b})$-matrix of level $\mathfrak{n}$. Applying the product rules of the previous Proposition, it is clear that for any $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix M, MB is a $W_{\mathfrak{q}}^{\mathfrak{m}_1}$-matrix of level $\mathfrak{n}$. Conversely, note that for any $W_{\mathfrak{q}}^{\mathfrak{m}_1}$-matrix M', $M = M'B^{-1}$ is a $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrix of level $\mathfrak{n}$, and so we can write $M' = MB$. □

Corollary 2.3.11. *For a fixed level $\mathfrak{n}$, the set of all $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices of level $\mathfrak{n}$ form a group $\mathcal{N}_W(\mathfrak{n})$ under multiplication modulo scalar factors. This group contains $\Gamma_0(\mathfrak{n})$ (modulo scalars) as a normal subgroup of finite index, and the quotient $\mathcal{N}_W(\mathfrak{n})/\Gamma_0(\mathfrak{n})$ is an elementary abelian 2-group.*

The elements of the quotient group $W(\mathfrak{n}) = \mathcal{N}_W(\mathfrak{n})/\Gamma_0(\mathfrak{n})$ induce involution operators on any space in which Γ acts. These involutions clearly generalise both the classical Atkin-Lehner involutions and the involutions coming from elements of Δ. The size of this group $W(\mathfrak{n})$ depends on the structure of the ideal class group and the ideal classes of the divisors of the level. For instance, when the class group has exponent 2 ($Cl \cong Cl[2]$ in the notation of §2.1) the only ideals in the trivial class are squares. In this case we have $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices only for $\mathfrak{q}$ principal, and $\mathfrak{m}$ is arbitrary. On the other hand, when the class number is odd every ideal class is a square and the situation is more similar to the classical one. We will have $W_{\mathfrak{q}}^{\mathfrak{m}}$-matrices for all $\mathfrak{q}||\mathfrak{n}$, and the ideal class of $\mathfrak{m}$ will be totally determined by $\mathfrak{q}$ (this case was discussed in M. Lingham's thesis [22, Chapter 5]).

Chapter 3

Geometry of the upper half space

In this chapter we restrict ourselves to the imaginary quadratic field case. For a field K with ring of integers R, we want to obtain a fundamental domain for the hyperbolic 3-space $\mathfrak{H}_3$ under the action of the group $\mathrm{GL}(2, R)$. From such a fundamental domain one can obtain a tessellation of $\mathfrak{H}_3$ by hyperbolic polyhedra. This tessellation contains all the geometric information about K necessary as input for the modular symbols algorithm.

This geometrical study goes back to work by Bianchi [4] in 1892, where for a few imaginary quadratic fields ($K = \mathbb{Q}(\sqrt{-d})$ with $d = 1, 2, 3, 5, 6, 7, 10, 11, 13, 15, 19$) he found fundamental domains for $\mathfrak{H}_3$ under the action of the so called *Bianchi groups*, $\mathrm{SL}(2, R)$. Following the work of Bianchi [4] and of Humbert [20], R. Swan [32] described a general method to determine fundamental domains for the action of Bianchi groups on $\mathfrak{H}_3$, which he used to give presentations for the Bianchi groups.

In §3.1 and §3.2 we recall the theory necessary to apply Swan's methods and obtain a fundamental domain for the action of $\mathrm{GL}(2, R)$. We also describe our implementation of the method, which in principle works for all imaginary quadratic fields without any restriction. To our knowledge, all previous implementations of the algorithms [22, 26] were dependent on estimates obtained by a preliminary partial run of the program. In §3.4 we describe a generalisation of the Euclidean algorithm which relies on the geometry of the fundamental domain. Next we review work of J. Bygott [6] which simplifies the geometry of the fundamental domain, especially for certain types of fields. In the last section we make a few comments about the next steps in the modular symbols algorithm for imaginary quadratic fields: finding

a tessellation for the hyperbolic 3-space and computing the rational 1-homology of $\Gamma_0(\mathfrak{n})\backslash\mathfrak{H}_3$.

3.1 The upper half space model

In the study of the three-dimensional hyperbolic space one may take different approaches, each with its own advantages (see [16, Chapter 1] for a discussion of several classical models). We use the upper half-space model since it resembles the upper half-plane as a model of plane hyperbolic geometry. Below we give a brief description of this model and some of its main properties which are of interest to us. More details can be found in [16, Chapters 1 and 2] and [6, Chapter 3].

Define the *upper half-space* as the set

$$\begin{aligned}\mathfrak{H}_3 = \mathbb{C}\times\mathbb{R}_{>0} &= \{(z,t):\, z\in\mathbb{C},\, t\in\mathbb{R}_{>0}\}\\ &= \{(x,y,t):\, x,y,t\in\mathbb{R},\, t>0\}.\end{aligned}$$

The space $\mathfrak{H}_3$ can be equipped with the hyperbolic metric coming from the line element:

$$ds^2 = \frac{dx^2+dy^2+dt^2}{t^2}$$

The induced topology is the Euclidean topology, but the geometry is hyperbolic. The geodesic lines are half-circles or half-lines in $\mathfrak{H}_3$ which are orthogonal to the boundary plane $\mathbb{C}$ in the Euclidean sense; the geodesic surfaces are Euclidean hemispheres or half-planes which again are orthogonal to the boundary $\mathbb{C}$.

The group $G = \mathrm{GL}(2,\mathbb{C})$ acts on the upper half space. This action can be described as follows:

$$\begin{aligned}\mathfrak{H}_3 &\longrightarrow \mathfrak{H}_3\\ (z,t) &\longmapsto M\cdot(z,t) = (z^*,t^*),\end{aligned}$$

where for any $M = \begin{pmatrix} a & b\\ c & d\end{pmatrix} \in G$, we have

$$z^* = \frac{(az+b)(\overline{cz+d}) + a\bar{c}t^2}{|cz+d|^2+|c|^2t^2}, \qquad t^* = \frac{|ad-bc|t}{|cz+d|^2+|c|^2t^2}\,. \tag{3.1.1}$$

In fact, G acts on $\mathfrak{H}_3$ as a group of isometries, that is, its action is compatible with the hyperbolic geometry on $\mathfrak{H}_3$.

Lemma 3.1.1. *The hyperbolic metric is invariant under the action of G. The set of geodesics for the hyperbolic metric is stable under the action of G.*

Proof. See [32, Lemmas 3.3 and 3.4, page 16]. □

We take the following definition for fundamental domains (as given in [16, §2.2]):

Definition 3.1.2. A closed subset $\mathcal{F}$ of $\mathfrak{H}_3$ is called a *fundamental domain* or *fundamental region* for the action of a discontinuous group G of isometries if it satisfies the following conditions:

(i) $\mathcal{F}$ meets each G-orbit at least once,

(ii) the interior of $\mathcal{F}$ meets each G-orbit at most once,

(iii) the boundary of $\mathcal{F}$ has Lebesgue measure zero.

In the next section we describe fundamental domains for certain subgroups of $G = \mathrm{GL}(2, \mathbb{C})$. These fundamental domains have the property of being bounded by finitely many geodesic surfaces. Hence it will be possible to give a more precise description of these particular domains. In fact, we have an algorithm (due to Swan [32]) that computes, for each subgroup, all the geometric information necessary to determine the corresponding fundamental region.

3.2 Fundamental domains for imaginary quadratic fields

Let K be an imaginary quadratic field with ring of integers R. Write $K = \mathbb{Q}(\sqrt{-d})$, where $d \in \mathbb{N}$ is square-free, and denote by D_K the discriminant of the field. Then $\{1, \omega\}$, with

$$\omega = \begin{cases} \frac{1}{2}\left(1 + \sqrt{-d}\right) & \text{if } d \equiv 3 \pmod 4 \\ \sqrt{-d} & \text{otherwise,} \end{cases} \tag{3.2.1}$$

is a basis of the ring of integers R ([27, §2.5 Theorem 1]).

Now let $\mathbb{P}^1(K)$ be the set of cusps for the field K (as in §1.2) and take $\Gamma = \mathrm{GL}(2, R)$. Recall (§1.2.5) that Γ acts transitively on the set $\mathbb{P}^1(K)$ only when $h_K = 1$, where h_K is the class number of K. Indeed, there are h_K orbits of cusps under the action of Γ (Proposition 1.2.24).

In this section we find a fundamental domain $\mathcal{F}_K$ for the action of Γ on the *extended upper half space* $\mathfrak{H}_3^*$, which we define by:

$$\mathfrak{H}_3^* = \mathfrak{H}_3 \cup \mathbb{P}^1(K) = \mathfrak{H}_3 \cup K \cup \{\infty\}.$$

3.2.1 Some geometry of imaginary quadratic fields

We need to describe a fundamental region F_K for the complex plane $\mathbb{C}$ with respect to the action of Γ_∞, the stabiliser of ∞ under Γ-action. This region F_K will appear in our latter description of the fundamental domain $\mathcal{F}_K$ for $\mathfrak{H}_3$. A nice and detailed explanation of how to obtain this fundamental region F_K can be found in J. Bygott's thesis [6, Chapter 3]. We briefly review some basic facts and introduce notation that will be useful in the rest of the chapter.

We consider some special elements of Γ. Define

$$T = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \quad U = \begin{pmatrix} \epsilon & 0 \\ 0 & 1 \end{pmatrix},$$

where ϵ is a generator of the unit group $R^\times$. That is,

$$\epsilon = \begin{cases} \sqrt{-1} & \text{if } d = 1, \\ (1+\sqrt{-3})/2 & \text{if } d = 3, \\ -1 & \text{otherwise.} \end{cases}$$

It is clear that $T^n = \begin{pmatrix} 1 & n \\ 0 & 1 \end{pmatrix}$ for $n \in \mathbb{Z}$. In general, for $a \in R$ we define:

$$T^a = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}.$$

These two elements, T^a and U, act both on $\mathfrak{H}_3$ and on the complex plane $\mathbb{C}$. The matrix T^a acts by translation and U acts by rotation:

$$\begin{aligned} &T^a \cdot z = z + a, \quad U \cdot z = \epsilon z, \quad \text{for } z \in \mathbb{C}; \\ &T^a \cdot (z,t) = (z+a,t), \quad U \cdot (z,t) = (\epsilon z, t), \quad \text{for } (z,t) \in \mathfrak{H}_3. \end{aligned}$$

It is not difficult to prove (see [6, Lemma 45]) that, up to multiplication by units, the stabiliser Γ_∞ is generated by U and the powers T^a:

$$\Gamma_\infty = R^\times \cdot \left\{ \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix} : a \in R^\times, b \in R \right\}.$$

For instance, when $d \neq 1, 3$, the stabilizer is given by

$$\Gamma_\infty = \left\{ \pm \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix} : a \in \{\pm 1\}, b \in R \right\}.$$

Hence to obtain the fundamental region F_K it is enough to cut into pieces the fundamental region of $\mathbb{C}$ with respect to translations (which is either a rectangle or an hexagon), where these "cuts" depend on the rotations given by the powers of U. Thus we obtain:

$$\begin{array}{rcll} F_K & = & \left\{x + y\sqrt{-d} : 0 \leq x \leq 1/2, -1/2 \leq y \leq 1/2\right\} & \text{if } d \neq 1,\ d \equiv 1, 2 \pmod 4, \\ F_K & = & \left\{x + y\sqrt{-d} : 0 \leq x \leq 1/2, -1/4 \leq y \leq 1/4\right\} & \text{if } d \neq 3,\ d \equiv 3 \pmod 4, \\ F_K & = & \left\{x + y\sqrt{-d} : 0 \leq x \leq 1/2, 0 \leq y \leq 1/2\right\} & \text{if } d = 1, \\ F_K & = & \{z \in \mathbb{C} : 0 \leq \Re(z) \leq 1/2, |\arg(z)| \leq \pi/6\} & \text{if } d = 3. \end{array} \tag{3.2.2}$$

Equivalently, we can describe F_K in the following way, which is slightly more convenient for practical implementations:

$$\begin{array}{rcll} F_K & = & \left\{x + iy \in \mathbb{C} : -1/2 < x \leq 1/2, 0 \leq y \leq \sqrt{|D_K|}/4\right\} & \text{if } d \neq 1, 3, \\ F_K & = & \{x + iy \in \mathbb{C} : 0 \leq x \leq 1/2, 0 \leq y \leq 1/2\} & \text{if } d = 1, \\ F_K & = & \left\{x + iy \in \mathbb{C} : 0 \leq x, \frac{\sqrt{3}}{3}x \leq y, y \leq \frac{\sqrt{3}}{3}(1 - y)\right\} & \\ & & \cup \left\{x + iy \in \mathbb{C} : 0 \leq x \leq \frac{1}{2}, -\frac{\sqrt{3}}{3}x \leq y \leq \frac{\sqrt{3}}{3}x\right\} & \text{if } d = 3. \end{array} \tag{3.2.3}$$

Remark 3.2.1. In our implementations we make use of the symmetry $z \mapsto \bar{z}$. This transformation is not in Γ, but it normalises Γ (i.e. its action commutes with that of Γ). In particular it allows us to obtain all the geometry from half of the region F_K.

3.2.2 Description of a fundamental domain for $\mathfrak{H}_3$ (theory of hemispheres)

A thorough account of the theory in this section and the next (§3.2.3) can be found in [16, Chapter 7], [32] or [6, Chapter 3]. Here we summarise the material more relevant for our (computational) purposes, trying to maintain our exposition as self-contained as possible. In the same spirit, we only include proofs when their strategies and ideas are useful later for our algorithms.

In general, the ring of integers R of a number field K is called *norm-Euclidean* if

its norm function is a Euclidean function (allowing to define a Euclidean algorithm). That is, R is Euclidean if for every $\beta \in K$ there exists $q \in R$ such that $\mathrm{N}(\beta - q) < 1$. In particular, we have the following geometrical interpretation of the Euclidean algorithm.

Algorithm 3.2.2 (Euclidean algorithm)**.** *Given elements $a, b \in R$, the algorithm finds a greatest common divisor of a and b.*

1. Translation step:

   If $b = 0$, return a.

   Otherwise, find $q \in R$ such that $\mathrm{N}(a/b - q) < 1$, and apply T^{-q} to $\binom{a}{b}$.

2. Inversion step:

   Apply the inversion matrix $S = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$ to $\binom{a}{b}$.

   Go to step 1.

Remark 3.2.3. Multiplying together all the translations and inversions that arise during the process, one obtains a matrix $M \in \mathrm{SL}(2, R)$ (note that $\det M = 1$) such that

$$M \binom{a}{b} = \begin{pmatrix} x & y \\ z & w \end{pmatrix} \binom{a}{b} = \binom{g}{0},$$

where $g = xa + yb$ is a greatest common divisor of a and b. We will generalise this process in section §3.4.

The algorithm terminates since the two steps reduce the size of the denominator $\mathrm{N}(b)$, and this can happen only finitely many times before b becomes zero (since $\mathrm{N}(b) \in \mathbb{N}$). The key property of the field on which the algorithm depends, the fact that it is Euclidean, can be interpreted geometrically: every $\beta \in K$ lies strictly inside a circle of radius one centred on an integral element.

From now we restrict ourselves again to the imaginary quadratic case, and we follow the notation introduced at the beginning of §3.2 (page 63). For the imaginary quadratic fields $K = \mathbb{Q}(\sqrt{-d})$ with $d = 1, 2, 3, 7, 11$ (R is not Euclidean for other values of d [17, §IV.1]), every $\beta \in K$ lies within at least one circle of radius one centred on an integral element $q \in R$. These radius 1 circles "cover the floor" of $\mathfrak{H}_3$. In Swan's paper [32] this idea is generalized to other values of d. Swan's strategy is to associate hemispheres to all principal cusps λ/μ, with radius that diminishes

as the size of the denominator $|\mu|$ grows. This class of hemispheres covers the floor if R is a principal ideal domain, and leaves out special isolated "singular points" which are not covered in the other cases. We now review this theory of hemispheres, which will lead to a very concrete description of a fundamental domain for $\mathfrak{H}_3^*$ under Γ-action.

The notion of a *pseudo-Euclidean* algorithm that substitutes the one above for non-Euclidean fields was first introduced in Elise Whitley's thesis [35]. Further work was done by Jeremy Bygott in his thesis [6], where he defined a *pseudo-Euclidean* function:

Definition 3.2.4. The *pseudo-Euclidean function* for K is the function given by:

$$\begin{array}{rcl} \psi : \mathbb{P}^1(K) & \longrightarrow & \mathbb{N} \cup \{0\} \\ \frac{\lambda}{\mu} & \longmapsto & \frac{\mathrm{N}\langle\mu\rangle}{\mathrm{N}\langle\lambda,\mu\rangle}. \end{array}$$

This definition does not depend on the representative chosen for the cusp. It is also clear that $\psi(\alpha) = 0$ if and only if $\alpha = \infty$, and in general $\langle\lambda,\mu\rangle|\langle\mu\rangle$ which implies that $\psi(\lambda/\mu) \in \mathbb{N}$. In fact, the pseudo-Euclidean function generalises the notion of "size of the denominator" by considering the norm of the corresponding denominator ideal, which we defined in §1.2.6 (Definition 1.2.25). Note that given a cusp $\alpha \in \mathbb{P}^1(K)$,

$$\psi(\alpha) = 1 \Leftrightarrow \alpha \in R,$$

and if $\alpha = \lambda/\mu$ is a principal cusp written in lowest terms, then $\psi(\alpha) = \mathrm{N}\langle\mu\rangle = |\mu|^2$.

Definition 3.2.5. For a cusp $\alpha \in K$, we define the *hemisphere* S_α *attached to* α as the set

$$S_\alpha = \left\{ (z,t) \in \mathfrak{H}_3 : |z-\alpha|^2 + t^2 = \frac{1}{\psi(\alpha)} \right\}.$$

With the hyperbolic metric, S_α is a geodesic surface (in Euclidean space it is a hemisphere). We will say that a point $(z,t) \in \mathfrak{H}_3$ lies under S_α, or that S_α covers the point (z,t), if

$$|z-\alpha|^2 + t^2 < \frac{1}{\psi(\alpha)}.$$

Definition 3.2.6. For a cusp $\alpha \in K$, we define the *circle* C_α *attached to* α as the set

$$C_\alpha = \left\{ z \in \mathbb{C} : |z-\alpha|^2 = \frac{1}{\psi(\alpha)} \right\}.$$

Clearly C_α is a circle in $\mathbb{C}$. In particular, if we identify $\mathbb{C}$ with the floor of the upper half space, $C_\alpha = \bar{S}_\alpha \cap \mathbb{C}$ (where $\bar{S}_\alpha$ denotes the closure of S_α). That is, C_α is the boundary of S_α regarded as a subset of $\mathfrak{H}_3 \cup \mathbb{P}^1(\mathbb{C})$. We define $S_\infty = \mathfrak{H}_3$ and $C_\infty = \mathbb{P}^1(\mathbb{C})$.

Definition 3.2.7. A hemisphere S_α or a circle C_α is called *principal* if α is a principal cusp.

Our definitions of hemispheres and circles attached to any cusp will be useful in §3.5. However in what follows we need only to consider principal hemispheres. Note that if $\alpha = \frac{\lambda}{\mu}$ is a principal cusp in lowest terms there is a simpler definition for S_α:

$$\begin{aligned} S_\alpha &= \left\{(z,t) \in \mathfrak{H}_3 : |z-\alpha|^2 + t^2 = \tfrac{1}{|\mu|^2}\right\} \\ &= \left\{(z,t) \in \mathfrak{H}_3 : |\mu z - \lambda|^2 + |\mu|^2 t^2 = 1\right\}. \end{aligned}$$

It is now obvious that a principal hemisphere S_α, with $\alpha = \lambda/\mu$, is an Euclidean hemisphere with centre at $(\lambda/\mu, 0)$ and radius $\frac{1}{|\mu|^2}$.

Lemma 3.2.8. *Let $(z,t) \in \mathfrak{H}_3$. The set of principal cusps α such that S_α covers (z,t) is finite.*

Proof. Let $\alpha = \frac{\lambda}{\mu}$ be a principal cusp such that S_α covers the point (z,t). Then the following inequality must hold:

$$|\mu z - \lambda|^2 + |\mu|^2 t^2 < 1. \tag{3.2.4}$$

In particular $|\mu|^2 t^2 < 1$, so we have the following upper bound for $|\mu|$:

$$|\mu|^2 < t^{-2}.$$

Since $\mu \in R$, the number of such elements μ is finite. By (3.2.4), each μ of this finite set satisfies

$$|\mu z - \lambda|^2 < 1 - |\mu|^2 t^2.$$

Write $\delta = \mu z - \lambda$. For each fixed μ, there are finitely many integral elements $\delta \in R$ satisfying $|\delta|^2 < 1 - |\mu|^2 t^2$. Hence there is only a finite number of possible choices for $\lambda = \mu z - \delta$. □

The proof of Lemma 3.2.8 contains all the ingredients to write an algorithm that returns the finite set of principal hemispheres covering a given point in the upper half space.

Algorithm 3.2.9. *Given a point (z,t) in $\mathfrak{H}_3$, the algorithm computes the list S of principal hemispheres that cover the point.*

1. Initialise S as an empty list.
2. Find $L_\mu = \{\mu \in R :\ \mathrm{N}(\mu) < 1/t^2\}$.
3. For each μ in L_μ:
   (a) Find $L_\lambda = \left\{\lambda \in R :\ \mathrm{N}(\lambda) < \left(\sqrt{1-|\mu|^2t^2} + |\mu z|\right)^2\right\}$.
   (b) For each λ in L_λ:
   - Check if λ/μ is a principal cusp. Note that it is enough to consider pairs (λ,μ) such that $\mathrm{N}(\langle\lambda,\mu\rangle) = 1$ (for any cusp generated by one of the pairs (λ,μ), its reduced representative is always in the list).
   - For pairs (λ,μ) satisfying the condition above, check if
   $$|\mu z - \lambda|^2 + |\mu|^2 t^2 < 1$$
   holds, and if that is the case, add the hemisphere $S_{\lambda/\mu}$ to the list S.

Following the strategy outlined in the proof Lemma 3.2.8, in step (a) we have used $|\lambda| \le |\mu z - \lambda| + |\mu z|$ and equation (3.2.4) to find a bound for the numerator λ.

Lemma 3.2.10. *Let $M_\alpha = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma$, $\alpha = -d/c$ a principal cusp and $(z',t') = M_\alpha \cdot (z,t)$. Then*
$$\frac{t}{t'} = |cz-d|^2 + |c|^2t^2.$$
Hence $t' > t$ if and only if (z,t) lies under the hemisphere S_α. Similarly, $t' = t$ if and only if $(z,t) \in S_\alpha$, and $t' < t$ if and only if (z,t) lies outside S_α.

Proof. By (3.1.1),
$$t' = \frac{t}{|cz+d|^2 + |c|^2t^2}.$$
For the special case $\alpha = \infty$, note that if $c = 0$, then $|d| = 1$ (since $M_\alpha \in \mathrm{GL}(2,R)$) and $t' = t$. The remarks at the end of the Lemma are clear if we recall that
$$S_\alpha = \{(z,t) \in \mathfrak{H}_3 : |cz+d|^2 + |c|^2t^2 = 1\}.$$
□

Lemma 3.2.10 shows that if a principal hemisphere S_α covers the point $(z,t) \in \mathfrak{H}_3$, then applying a matrix M_α raises the "height" of (z,t). This will happen only finitely many times, since by Lemma 3.2.8 only finitely many greater heights can be obtained (there is only a finite number of principal hemispheres covering a given point of $\mathfrak{H}_3$). The points which cannot be raised, because they do not lie under any suitable hemisphere, are of special importance.

Definition 3.2.11. Define $\mathcal{B}_k$ as the set of points $(z,t) \in \mathfrak{H}_3$ that lie above all principal hemispheres S_α:

$$\mathcal{B}_K = \left\{(z,t) \in \mathfrak{H}_3 : |cz-d|^2 + |c|^2 \geq 1 \text{ for all } c,\, d \in R \text{ with } \langle c,d\rangle = R\right\}.$$

We will call its boundary, $\partial\mathcal{B}_K$, the *Bianchi diagram* for Γ (think of it as the "floor" of the region $\mathcal{B}_K$).

Theorem 3.2.12. *Let F_K be a fundamental region for $\mathbb{C}$ with respect to Γ_∞, such as the one described above in* (3.2.2) *or* (3.2.3)*. Then the set*

$$\mathcal{F}_K = \{(z,t) \in \mathcal{B}_K : z \in F_K\}$$

is a fundamental region for Γ on $\mathfrak{H}_3$.

Proof. [6, Theorem 52], [16, Chapter 7] or [32, §3]. □

Remark 3.2.13. Part of the proof of Theorem 3.2.12 gives a method to send points of $\mathfrak{H}_3$ to the fundamental region $\mathcal{F}_K$. This is easily made into an algorithm as we will see in §3.4 (Algorithm 3.4.2). Here we sketch this part of the proof. Let (z,t) be any point in $\mathfrak{H}_3$. If no principal hemisphere covers the point it is clear that $(z,t) \in \mathcal{B}_K$. Otherwise, among the finitely many principal cusps $\alpha = \lambda/\mu$ such that S_α covers (z,t) we choose one that minimises the quantity $|\mu z - \lambda|^2 + |\mu|^2 t^2$. Now for this hemisphere S_α we construct a matrix M_α such that $M_\alpha \cdot \alpha = \infty$ (as in Lemma 3.2.10), and put $(z',t') = M_\alpha \cdot (z,t)$. Multiplying M_α on the left by a convenient element of the stabilizer Γ_∞ we can assume that $z' \in \bar{F}_K$. By Lemma 3.2.10, (z',t') has maximal t-coordinate among all other points in the Γ-orbit of (z,t), so no principal hemisphere can cover (z',t'). Hence $(z',t') \in \mathcal{F}_K$.

These matrices M_α associated to principal cusps α will have some importance in our computations. We explain how to construct them explicitly in §3.4.

Theorem 3.2.12 gives a very concrete description of a fundamental domain for Γ. Now some more work on this theory of hemispheres will allow us to improve our understanding of the geometry of $\mathcal{F}_K$ and also lay the basis for a pseudo-Euclidean

algorithm on R. Consider the action of matrices M_α (as in Lemma 3.2.10) on cusps instead of interior points:

Lemma 3.2.14. *Let* $M_\alpha = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Gamma$, $\alpha = -d/c$ *a principal cusp and* $\beta \in \mathbb{P}^1(K)$ *an arbitrary cusp. Then*

$$\psi(M_\alpha \cdot \beta) = \begin{cases} \psi(\beta) & \text{if } \alpha = \infty, \\ |c|^2 & \text{if } \beta = \infty, \\ \psi(\beta)|c\beta + d|^2 & \text{otherwise.} \end{cases} \tag{3.2.5}$$

Hence $\psi(M_\alpha \cdot \beta) < \psi(\beta)$ *if and only if* β *lies inside* C_α. *Similarly,* $\psi(M_\alpha \cdot \beta) = \psi(\beta)$ *if and only if* $\beta \in C_\alpha$, *and* $\psi(M_\alpha \cdot \beta) > \psi(\beta)$ *if and only if* β *lies outside* C_α.

Proof. We follow [6, Lemma 50], where this result is proved for a slightly more general case. Write $\beta = \lambda/\mu$, and let $\lambda' = a\lambda + b\mu$, $\mu' = c\lambda + d\mu$, so that $M_\alpha \cdot \beta = \lambda'/\mu'$. By Lemma 1.2.23, $\langle \lambda', \mu' \rangle = \langle \lambda, \mu \rangle$. Therefore:

$$\psi(M_\alpha \cdot \beta) = \frac{\mathrm{N}\langle \mu' \rangle}{\mathrm{N}\langle \lambda', \mu' \rangle} = \frac{|c\lambda + d\mu|^2}{\mathrm{N}\langle \lambda, \mu \rangle} = \frac{\mathrm{N}\langle \mu \rangle |c\beta + d|^2}{\mathrm{N}\langle \lambda, \mu \rangle}.$$

Now recall that we have defined $C_\infty = \mathbb{P}^1(\mathbb{C})$, which has neither an inside nor an outside, and so the concluding remarks of the Lemma hold trivially in the case $\alpha = \infty$. Otherwise they follow from (3.2.5). □

From the Lemma it is clear that if a cusp β lies inside a principal circle C_α, then applying the associated matrix M_α reduces the "size" of the cusp as measured by ψ. In view of this, it will be convenient to characterize the cusps which lie inside no circle C_α.

Definition 3.2.15. A point $z \in \mathbb{C}$ is *singular* if it lies inside no principal circle C_α. That is, if $|\mu z - \lambda| \geq 1$ for all principal cusps $\alpha = \lambda/\mu$.

Lemma 3.2.14 provides a characterization for singular cusps (points of K which are singular) as minimal elements with respect to the pseudo-Euclidean function:

Corollary 3.2.16. *Let* $\beta \in K$. *Then the cusp* β *is singular if and only if* $\psi(\beta)$ *is minimal for points in the* Γ*-orbit of* β.

Proof. Straightforward from Lemma 3.2.14 (see details in [6, Corollary 51]). □

Remark 3.2.17. We note that there are no singular cusps when K has class number one. This could be deduced from Corollary 3.2.16: all cusps are in the same Γ-orbit and the point with minimal ψ in the orbit is the cusp ∞ ($\psi(\infty) = 0$), which by definition is not singular. Also it is clear from Definition 3.2.15 that principal cusps cannot be singular, since any principal cusp $\alpha \neq \infty$ will trivially lie inside the principal circle C_α.

It can be shown, using Diophantine approximation, that in fact singular points lie necessarily in K (see Theorem 3.2.19 below). Thus from now on we will not make any distinction between the terms *singular point* and *singular cusp*. In §3.2.3 we will discuss these special points in more detail, and we will see that in fact there are only finitely many singular cusps modulo translation by elements of R. We conclude this section with a finiteness result for our fundamental domains, which is proved using Diophantine approximation techniques and properties of singular points.

Theorem 3.2.18. *The boundary of $\mathcal{F}_K$ is defined by finitely many geodesic surfaces.*

Proof. [16, Chapter 7, Theorem 3.8] or [32, Theorem 3.13]. □

Our next objective is to describe a method for finding the principal hemispheres on the boundary of $\mathcal{B}_K$ (the Bianchi diagram for Γ) over F_K, of which there are only a finite number (by Theorem 3.2.18). The next two sections will cover this topic.

3.2.3 Singular points

We want to determine the cell structure of the floor $\partial\mathcal{B}_K$ of the fundamental domain $\mathcal{F}_K$. The method we will describe in §3.2.4 requires a list of singular points which must be computed in advance. Thus it is convenient to be able to identify singular points. Note that the characterization of singular cusps given by Corollary 3.2.16 is not a practical one (from a computational point of view). We now review the basic properties of singular points and some results in Swan's paper [32, §7], where a method to enumerate singular cusps is described.

Theorem 3.2.19. *The singular points all lie in K. There are finitely many points $s_1, s_2, \ldots, s_r$ such that the singular points are exactly $s_i + \omega$, for $i = 1, \ldots, r$, $\omega \in R$.*

Proof. We follow the proof in [16, Chapter 7, Proposition 2.9] (the same proof is given in [32, Proposition 3.11]). For $z \in \mathbb{C}$, if $z \notin K$, using diophantine approximation ([16, §7.2. Proposition 2.7] or [32, Theorem 2.1]) we can find $\lambda, \mu \in R$ such that $\langle \lambda, \mu \rangle = R$ and $|z - \frac{\lambda}{\mu}| \leq 1$, so in particular z is not singular.

Choose $\mathfrak{a}_0 = R, \mathfrak{a}_1, \ldots, \mathfrak{a}_{h-1}$ representatives for the ideal classes of K, and for $i = 1, \ldots, h-1$ define

$$M_i = \{\gamma \in \mathfrak{a}_i \setminus \{0\} : |\gamma| \leq |\lambda| \text{ for all } \lambda \in \mathfrak{a}_i\},$$

the (finite and non-empty) set of nonzero minimal elements in the ideal $\mathfrak{a}_i$.

Let s be a singular point. Then $s \in K$ and we can write $s = \frac{\alpha}{\beta}$, with $\alpha, \beta \in R$ and $\langle \alpha, \beta \rangle = \mathfrak{a}_i$ for some $i \in \{1, \ldots, h-1\}$ (recall from Remark 3.2.17 that s cannot be a principal cusp). Now take $\gamma \in M_i$; there exists $\delta \in \mathfrak{a}_i$ with $\langle \gamma, \delta \rangle = \mathfrak{a}_i = \langle \alpha, \beta \rangle$. In particular we can find $\lambda, \mu \in R$ such that $\langle \lambda, \mu \rangle = R$ and $\mu\alpha + \lambda\beta = \gamma$. Since s is a singular point, it follows that $|\mu s + \lambda| \geq 1$. On the other hand, $|\mu s + \lambda| = |\gamma|/|\beta|$ so that we have $|\beta| \leq |\gamma|$ with γ a minimal element. Thus $\beta \in M_i$.

This proves that any singular point s is of the form $s = \frac{\alpha}{\beta}$ with β minimal in the ideal $\langle \alpha, \beta \rangle = \mathfrak{a}$. There is only a finite number of possibilities for β. And for each β, $\mathfrak{a}/\langle \beta \rangle$ is finite so there are only a finite number of choices for α modulo β. Clearly for any $\omega \in R$, s is singular if and only if $s + \omega$ is, and the second statement of the Theorem follows. □

Corollary 3.2.20. *Let $\mathfrak{a}_0 = R$, $\mathfrak{a}_1$, …, $\mathfrak{a}_{h-1}$ be a set of representatives for the ideal classes of K. For each $i \in \{1, \ldots, h-1\}$, write $\mathfrak{a}_i = \langle \beta, \alpha \rangle$ in all possible ways with β minimal in $\mathfrak{a}_i$, and form the cusp α/β. In this way we get all possible singular points or cusps, each exactly twice, once as α/β and once as $(-\alpha)/(-\beta)$.*

Proof. It follows from the proof of Theorem 3.2.19. See [32, Proposition 7.1] for details. □

Remark 3.2.21. Let h_K be the class number of K. We already know that there are no singular points when $h_K = 1$. Now Corollary 3.2.20 gives information when R is not a principal ideal domain: if $h_K > 1$, there are at least $h_K - 1$ singular points (modulo R).

We have now a method to list all singular points: enumerate representatives for all ideal classes and determine their minimal elements. In his paper, Swan follows this approach to give an explicit construction of a set of representatives for the singular points of an imaginary quadratic field.

Proposition 3.2.22. *The singular points of $K = \mathbb{Q}(\sqrt{-d})$, modulo R, are given by $\frac{p(r+\sqrt{-d})}{s}$, where $r, s \in \mathbb{Z}$ are such that*

$$s > 0, \quad -s/2 < r \leq s/2, \quad s^2 \leq r^2 + d,$$

and

(i) *If $d \not\equiv 3 \pmod 4$: $s|r^2 + d$, $s \neq 1$, $\gcd(p, s) = 1$ and p is taken modulo s.*

(ii) *If $d \equiv 3 \pmod 4$: s is even, $s \neq 2$, $2s|r^2 + d$, $\gcd(p, s/2) = 1$ and p is taken modulo $s/2$.*

Proof. [32, §7, pages 43-48]. □

Thus we have a straightforward method to enumerate representatives for singular points of imaginary quadratic fields. For example, take $d = 23$. Since $h_K = 3$ we know there are at least 2 classes of singular points, and following the proposition we obtain the set of representatives $\{\frac{1}{4}\left(\pm 1 + \sqrt{-23}\right)\}$.

Remark 3.2.23. Note that the representatives of singular points we obtain with the method of the Proposition above are not necessarily of smallest norm. In our algorithm for finding the fundamental domain $\mathcal{F}_K$ we will be interested only in the singular points which lie inside the fundamental rectangle F_K, and so it will be necessary to check if translations (by elements of R) of the representatives found are in F_K. For instance, the construction of Proposition 3.2.22 yields the following set of representatives for the singular points of $K = \mathbb{Q}(\sqrt{-91})$ (in this case $h_K = 2$):

$$\left\{ \frac{\pm 3 + \sqrt{-91}}{10}, \frac{\pm 3 + \sqrt{-91}}{5}, \frac{\pm 9 + 3\sqrt{-91}}{10}, \frac{\pm 6 + 2\sqrt{-91}}{10} \right\}.$$

But the set of singular points which lie inside the rectangle F_K described by (3.2.3) is

$$\left\{ \frac{1}{5}\left(\pm 2 + \sqrt{-91}\right), \frac{1}{10}\left(\pm 3 + \sqrt{-91}\right) \right\},$$

where $\frac{1}{5}(2 + \sqrt{-91}) = \frac{1}{5}(-3 + \sqrt{-91}) + 1$ and $\frac{1}{5}(-2 + \sqrt{-91}) = \frac{1}{5}(+3 + \sqrt{-91}) - 1$.

3.2.4 Finding the floor of the fundamental region

We will now describe a method for finding the Bianchi diagram for Γ, that is, the finite set of hemispheres that define the floor of $\mathcal{F}_K$. For the sake of simplicity, from now when we use the notation $\mathcal{B}_K$ we will mean only the part of $\mathcal{B}_K$ that lies over F_K.

Our method has two main steps. First we need to find a set S_0 of principal hemispheres that covers the fundamental rectangle F_K, except for finitely many isolated points (points corresponding to singular cusps).

In order to find this initial covering, we will start with the hemisphere with the biggest possible radius, $\alpha_1 = 0/1$, and then keep adding principal hemispheres $\alpha_i = \lambda_i/\mu_i$ with increasing $|\mu_i|$. We know that this process will terminate after a finite number of steps (by Theorem 3.2.18).

Remark 3.2.24. Although Theorem 3.2.18 guarantees that we have only to consider a finite number of possibilities for $|\mu|$ when creating our initial list of hemispheres, it does not give an explicit bound. On the other hand, as shown by Swan [32, §8], it is not difficult to calculate an upper bound for $|\mu|$. However, the bound obtained is far too large and therefore impractical for computational purposes. Take for instance the field $K = \mathbb{Q}(\sqrt{-43})$; since $h_K = 1$, we do not have singular points and it is easier to obtain a bound for the size of μ. In general, for the field $K = \mathbb{Q}(\sqrt{-d})$, with discriminant D_K and $\{1, \omega\}$ a basis for its ring of integers R, we define:

$$D = \begin{cases} \frac{1}{2}\sqrt{D_K + 4}, & \textit{if } d \equiv 1, 2 \pmod 4, \\ \\ \frac{1}{2}\sqrt{D_K + 9}, & \textit{if } d \equiv 3 \pmod 4, \end{cases}$$

and

$$B_{max} = D\sqrt{D_K/3} + \frac{1}{2}\sqrt{D_K/3} + |\omega|.$$

Let U be the union of a convenient set of neighbourhoods of the singular points. Then for $M \in \mathbb{Z}$ large enough so that $DB_{max}/M < 1/4$, it can be proved (by fairly straightforward arguments that use diophantine approximation and are explained with detail in Swan's paper [32]) that all principal hemispheres $S_{\lambda/\mu}$ that meet the fundamental region $\mathcal{F}_K$ outside U, satisfy $|\mu| \leq 2B_{max}DM/\sqrt{3}$.

In the case $d = 43$ there are no singular points, so according to the statement above it will be enough to consider all μ satisfying $|\mu| \leq 2B_{max}DM/\sqrt{3} \approx 21\,436$ to guarantee that we find the principal hemispheres $S_{\lambda/\mu}$ that form the floor of $\mathcal{F}_K$. In fact, the smallest hemisphere in $\mathcal{F}_K$ for this field satisfies $|\mu| = 3$, which illustrates the inefficiency of the bound.

Because of this difficulty in bounding the size of $|\mu|$ it is necessary to use a different strategy in our algorithm for finding an initial covering. We will discuss this and other practical aspects of the computations in the next section.

The second part of the method is to calculate the floor of $\mathcal{F}_K$ from our initial list of hemispheres. The following result (by Swan) explains how to do this.

Proposition 3.2.25. *Let $S_{\alpha_1}, \ldots, S_{\alpha_n}$ be a set of principal hemispheres that covers F_K (except for a finite number of isolated points), and define*

$$\mathcal{B}(\alpha_1, \ldots, \alpha_n) = \{(z,t) \in \mathfrak{H}_3 : (z,t) \text{ lies over all the hemispheres } S_\alpha, \ldots, S_{\alpha_n}\}$$

Then $\mathcal{B}(\alpha_1, \ldots, \alpha_n) = \mathcal{B}_K$ if and only if no vertex of $\partial\mathcal{B}(\alpha_1, \ldots, \alpha_n)$ (that is, any point of intersection of three or more hemispheres) can be covered by any principal hemisphere.

Proof. [32, Proposition 8.4] □

Algorithm 3.2.26 (Swan's algorithm)**.** *Given an imaginary quadratic field K and $S_0 = \{S_{\alpha_i} : i = 1, \ldots, n\}$ an initial covering of the fundamental rectangle F_K as the one described in* Proposition 3.2.25*, this algorithm returns a list S of the principal hemispheres which form the Bianchi diagram for Γ.*

1. Set $S = S_0$.
2. Find $V = \{v \in \mathfrak{H}_3 : v \in S_\alpha \text{ for 3 or more different } S_\alpha \in S\}$.
3. For each $v \in V$: using Algorithm 3.2.9, compute the finite set
$$S_v = \{\text{principal hemispheres that cover } v\}.$$
4. If $S_v = \emptyset$ for all $v \in V$, return S.

   Otherwise,

   (a) For each $v = (z,t) \in V$ with $S_v \neq \emptyset$:
      (i) Find S_β with $(z,t') \in S_\beta$ that has the largest t coordinate (i.e. S_β is the hemisphere covering v with maximum "height").
      (ii) If $S_\beta \notin S$, set $S = S \cup \{S_\beta\}$.

   (b) Go to step 2.

Note the importance of Lemma 3.2.8. It guarantees that the set S_v of step 3 in the algorithm is finite, and its proof led to Algorithm 3.2.9, which we use to compute S_v.

3.2.5 Implementation of the algorithms

In this section we discuss some of the practical aspects of the implementation of the algorithms described above. In principle our program (written in *Sage* [31]) works for all imaginary quadratic fields $K = \mathbb{Q}(\sqrt{-d})$, with d a square-free positive integer such that $d \neq 1, 3$. A printed version of the source code can be found in Appendix A.2.

First of all we observe that thanks to the symmetry given by $z \mapsto \bar{z}$ (recall Remark 3.2.1) it is enough for our computations to consider only half of the region F_K. For simplicity we keep the notation F_K for this new reduced fundamental rectangle. Throughout the rest of the section, F_K is defined by

$$F_K = \left\{ x + iy \in \mathbb{C} : 0 < x \leq 1/2,\, 0 \leq y \leq \sqrt{|D_K|}/4 \right\}. \tag{3.2.6}$$

We need to create an initial list of principal hemispheres that cover the rectangle F_K with the exception of a finite set of isolated points. Now we recall our strategy, which was outlined at the beginning of the previous section (page 74):

1. initialise the list with all principal hemispheres of radius one (the biggest possible radius) which intersect F_K,
2. keep adding principal hemispheres $\alpha_i = \lambda_i/\mu_i$, with increasing $|\mu_i|$, until F_K is covered (up to a finite number of isolated points).

Theorem 3.2.18 guarantees that we need only to consider a finite number of possibilities for $|\mu|$ when creating this initial list of hemispheres, and so we will be done in a finite number of steps. However, we do not know beforehand how many steps are needed (Remark 3.2.24). Previous implementations (see [22] and [26]) seem to rely on estimates obtained from a preliminary run of the program. Our approach was to implement a function that tests if a given rectangle on the floor of $\mathfrak{H}_3$ is totally covered by a given set of hemispheres.

The test we designed is a breadth-first recursive function, which not only checks the covering of a given list of rectangles but it also adds new hemispheres to our initial list until we are certain that everything is covered. It works as follows:

Algorithm 3.2.27. *Let S_0 be a list of principal hemispheres and $R = \{R_1, \ldots, R_s\}$ a collection of rectangles in F_K which we want to cover with the hemispheres in S_0. The algorithm adds new hemispheres to the list S_0 until all rectangles in R are covered.*

1. If the set R is empty, we are done.
2. For each R_i in R we check the covering of its four vertices as follows:
   (a) If the four vertices of R_i are covered by the same hemisphere of S_0: remove R_i from the list R.
   (b) If the four vertices of R_i are covered by hemispheres of S_0 but they are not inside the same one:

       Subdivide R_i in smaller rectangles $\{R_{i_1}, R_{i_2}, \ldots, R_{i_k}\}$.

       In the list R replace the rectangle R_i by the collection of its sub-rectangles.
   (c) If at least one vertex of R_i is not covered: add hemispheres for the next value of $|\mu|$ to S_0. Repeat until all four vertices of R_i are covered by hemispheres of S_0. Then:

       If all four vertices of R_i are covered by the same hemisphere, remove R_i from the list R.

       Otherwise, in the list R replace R_i by the corresponding sub-rectangles as we did in step (b).
3. Go to step 1.

The obvious approach is then to apply this function to the whole fundamental rectangle F_K minus the singular points. However, when we are close to a singular cusp the test function is extremely inefficient. Recall that the singular cusps do not lie under any hemisphere. In particular, when any of the four vertices of a rectangle is in the neighbourhood of a singular point to find that all four vertices are covered by the same hemisphere we need a very small rectangle. This leads to a very long recursion (too many subdivisions of the rectangle are needed to obtain a final answer).

On the other hand, given a singular cusp it is easy to check if it is totally surrounded by hemispheres and if that is the case, a straightforward computation will find a rectangular neighbourhood of the singular cusp which is totally covered by the surrounding hemispheres. That led us to treat the singular points separately

and in advance, using Proposition 3.2.22 to enumerate all singular cusps beforehand. Our final implementation follows the algorithm below.

Algorithm 3.2.28 (Finding an initial covering of F_K)**.** *Given an imaginary quadratic field K, the algorithm returns a list of principal hemispheres that cover the fundamental rectangle F_K (except singular points).*

1. Initialize $S_0 = \{S_{0/1}, S_{1/1}, S_{\omega/1}\}$, where $\{1, \omega\}$ is a basis for the ring of integers of K, as given by (3.2.1). This is the set of hemispheres of radius 1 that intersect F_K.

2. Compute the set of singular points $\mathcal{S}_K = \{s_1, \ldots, s_r\}$ which are inside the rectangle F_K.

3. Add hemispheres (increasing values of $|\mu|$) to our initial set S_0 until all s in $\mathcal{S}_K$ are totally surrounded by hemispheres.

4. For each s_i in $\mathcal{S}_K$: find $R(s_i)$, a rectangular neighbourhood of s_i which is totally covered by the hemispheres surrounding s_i.

5. Consider the rectangle F_K minus the neighbourhoods $R(s_i)$ of singular points:

$$D = F_K \setminus \bigcup_{i=1}^{r} R(s_i).$$

   Subdivide the region D in rectangles $R = \{R_1, \ldots, R_s\}$.

6. Apply Algorithm 3.2.27 to the list R of rectangles.

As mentioned above, steps 3 and 4 are not too difficult. In particular to check if a singular cusp $s \in \mathcal{S}_K$ is surrounded by hemispheres in all directions we need only to consider the situation on the floor: we must check if our singular point is surrounded by principal circles C_α in all possible directions (where $C_\alpha = \overline{S}_\alpha \cap \mathbb{C}$ for S_α in our list of hemispheres). We take all principal circles C_α through s and compute the slopes of the corresponding tangent lines through the point. From each slope we find a circular sector in which the neighbourhood of s is covered. If the union of the central angles of all our sectors is $[0, 2\pi]$ we know that s is totally surrounded.

For the construction of a rectangular neighbourhood of $s \in \mathcal{S}_K$ which we can assure is totally covered by the corresponding surrounding hemispheres we proceed as follows.

Algorithm 3.2.29. *Let S be a list of principal hemispheres. Given a singular point $s \in \mathcal{S}_K$ which is totally surrounded by a subset of the hemispheres in S, the algorithm returns a rectangle around s which is completely covered.*

1. Initialize $C_0 = \{C_\alpha : s \in C_\alpha\}$, where $C_\alpha = \overline{S}_\alpha \cap \mathbb{C}$ for $S_\alpha \in S$.
2. Find $V = \{v : v = C_{\alpha_i} \cap C_{\alpha_j}, \text{ for } C_{\alpha_i}, C_{\alpha_j} \in C_0\}$.
3. Find $v_0 \in V$ at minimum distance from the singular point s.
4. Output the rectangle with centre at s and one vertex at v_0.

In section §3.2.4, after finding an initial set of hemispheres S_0 covering F_K, we said we may proceed to apply Swan's algorithm to determine the floor of $\mathcal{B}_K$. In practice it is convenient to clean our initial list of redundant hemispheres first. By a redundant hemisphere we mean any hemisphere which is in fact totally covered by others hemispheres in S_0 (recall that the floor of $\mathcal{B}_K$ consists of the highest hemispheres covering F_K). To check if a given hemisphere S_α in $S_0 = \{S_{\alpha_1}, \ldots, S_{\alpha_n}\}$ is totally covered we begin by computing all its intersection points with 2 or more other hemispheres:

$$P_\alpha = \{p : p \in S_\alpha \text{ and } p \in S_{\alpha_i} \text{ for 2 or more } i \in \{1, \ldots, n\}\}.$$

Then if all points p in P_α are strictly covered by other hemispheres in S_0, we can assure that S_α is totally covered and it can removed from S_0.

The implementation of Swan's algorithm (Algorithm 3.2.26) is straightforward. The most complex part of that computation comes in step 3, in which we find the set S_v of principal hemispheres that cover the point v. As mentioned previously, the proof of Lemma 3.2.8 gives us a method to find v: we need only to check a finite number of principal hemispheres $S_{\lambda/\mu}$, where $|\lambda|, |\mu|$ satisfy the bounds given in the proof. Unfortunately this process may take a rather long time. For instance, if no principal hemisphere covers v and there is a large number of choices for λ and μ, then we must check all possibilities before giving a definite answer.

It is worth observing that when the full computation to determine the fundamental region $\mathcal{F}_K$ has been made once it is possible to repeat it faster. That is because if we know beforehand the radius of the smallest hemisphere in the floor of $\mathcal{B}_K$, then we can use this knowledge to efficiently bound μ in all the algorithms (including Swan's). Using this pre-computed bound in our routine for Swan's algorithm approximately doubles the speed of the computation.

3.3 Examples

The following tables summarise the output of our program for a number of imaginary quadratic fields $K = \mathbb{Q}(\sqrt{-d})$. The tables do not include the complete set of hemispheres which defines the fundamental domain $\mathcal{F}_K$, since it can be quite long (especially as the discriminant of the field increases). We denote the discriminant of the field by D_K. All our tables give the maximum size of denominators needed to form the fundamental domain:

$$\max\left\{\mathrm{N}(\mu) = |\mu|^2 : \text{ for } S_{\lambda/\mu} \in \partial\mathcal{B}_K\right\},$$

(recall that this is the reciprocal of the square of the radius of the smallest hemisphere in the Bianchi diagram). Also included is the square of the height of the lowest vertex of $\mathcal{F}_K$, excluding singular points which have height zero. This last number appears under the heading "min t^2". Below we list this data for all imaginary fields with class number 1 (Table 3.1). For completeness we include the fields with $d = 1$ and $d = 3$, whose fundamental domains have been computed many times in the literature (see for instance [4, 32, 10, 16]). For the eighteen fields of class number $h_K = 2$ (Table 3.3), we also give the list $\mathcal{S}_K$ of singular points in the fundamental rectangle. In Tables 3.2 and 3.4 we show results for fields with bigger class number ($h_K > 2$); we give the structure of the class group (under the heading "Cl_K") but not the singular points (the list can get rather long for big discriminants).

Remark 3.3.1. The values for "min t^2" in our tables agree with the corresponding results listed in Riley's paper [26, Table 1] (under the heading "RSHLV" Riley gives the value of $1/(\min t^2)$ for each case). The computations in that paper include the fundamental domains for all fields of class number one and fields $K = \mathbb{Q}\left(\sqrt{-d}\right)$ with $d \leq 37$.

Furthermore, the output of our program provides all the information we need to draw fundamental regions (a list of the hemispheres forming the Bianchi diagram for each field). We wrote a routine in *Sage* to plot the fundamental regions. Here we give two examples, for $K = \mathbb{Q}(\sqrt{-39})$ (class number 4) and $K = \mathbb{Q}(\sqrt{-47})$ (class number 5). Figures 3.1 and 3.3 show the corresponding Bianchi diagrams, and in Figures 3.2 and 3.4 we can see their projection on the floor, with points marking the projections of the vertices (i.e. singular cusps and points in the intersection of 3 or more hemispheres).

d	D_K	max $\lvert\mu\rvert^2$	min t^2
1	-4	1	1/2
2	-8	1	1/4
3	-3	1	2/3
7	-7	1	3/7
11	-11	1	2/11
19	-19	4	2/19
43	-43	9	2/43
67	-67	23	2/67
163	-163	53	2/163

Table 3.1: Fields $K = \mathbb{Q}(\sqrt{-d})$ with $h_K = 1$

d	D_K	Cl_K	max $\lvert\mu\rvert^2$	min t^2
14	-56	C_4	56	1/108
17	-68	C_4	68	3/400
21	-84	$C_2 \times C_2$	84	1/192
23	-23	C_3	16	11/184
31	-31	C_3	20	2/49
39	-39	C_4	39	1/49
47	-47	C_5	36	1/49
55	-55	C_4	56	701/59895
59	-59	C_3	36	55/2124
71	-71	C_7	64	55/5112
79	-79	C_5	64	551/51192
83	-83	C_3	36	1/49
87	-87	C_6	88	1/121
95	-95	C_8	95	1/121

Table 3.2: All fields $K = \mathbb{Q}(\sqrt{-d})$ with $h_K > 2$ and discriminant $|D_K| < 100$

d	D_K	max $\lvert\mu\rvert^2$	min t^2	$\mathcal{S}_K$
5	-20	20	1/25	$\{\pm 1/2 + 1/2\sqrt{-d}\}$
6	-24	24	3/100	$\{1/2\sqrt{-d}\}$
10	-40	44	3/196	$\{1/2\sqrt{-d}\}$
13	-52	52	3/256	$\{\pm 1/2 + 1/2\sqrt{-d}\}$
15	-15	15	3/49	$\{\pm 1/4 + 1/4\sqrt{-d}\}$
22	-88	88	3/676	$\{1/2\sqrt{-d}\}$
35	-35	35	3/121	$\{\pm 1/6 + 1/6\sqrt{-d}\}$
37	-148	289	3/1600	$\{\pm 1/2 + 1/2\sqrt{-d}\}$
51	-51	51	1/64	$\{\pm 1/2 + 1/6\sqrt{-d}\}$
58	-232	729	3/3844	$\{1/2\sqrt{-d}\}$
91	-91	91	3/289	$\{\pm 2/5 + 1/5\sqrt{-d},\ \pm 3/10 + 1/10\sqrt{-d}\}$
115	-115	115	1/144	$\{\pm 1/2 + 1/10\sqrt{-d},\ 1/5\sqrt{-d}\}$
123	-123	123	3/625	$\{\pm 1/2 + 1/6\sqrt{-d}\}$
187	-187	187	3/625	$\{\pm 5/14 + 3/14\sqrt{-d}\ ,\ \pm 3/14 + 1/14\sqrt{-d},\ \pm 3/7 + 1/7\sqrt{-d}\}$
235	-235	235	1/324	$\{\pm 1/2 + 1/10\sqrt{-d},\ 1/5\sqrt{-d}\}$
267	-267	400	3/2401	$\{\pm 1/2 + 1/6\sqrt{-d}\}$
403	-403	403	3/1225	$\{\pm 2/11 + 1/11\sqrt{-d},\ \pm 4/11 + 2/11\sqrt{-d},\ \pm 9/22 + 1/22\sqrt{-d},\ \pm 5/22 + 3/22\sqrt{-d},\ \pm 1/22 + 5/22\sqrt{-d}\}$
427	-427	427	1/576	$\{\pm 1/2 + 1/14\sqrt{-d}\ ,\ 1/7\sqrt{-d},\ \pm 1/2 + 3/14\sqrt{-d}\}$

Table 3.3: Fields $K = \mathbb{Q}(\sqrt{-d})$ with $h_K = 2$

d	D_K	Cl_K	max $\lvert\mu\rvert^2$	min t^2
26	-104	C_6	121	1/300
29	-116	C_6	169	3/1024
30	-120	$C_2 \times C_2$	169	3/1156
33	-132	$C_2 \times C_2$	225	87/94864
34	-136	C_4	225	3/1444
38	-152	C_6	289	1/588
41	-164	C_8	361	3/1936
42	-168	$C_2 \times C_2$	361	105/139876
46	-184	C_4	441	3/2500
53	-212	C_6	625	3/3136
57	-228	$C_2 \times C_2$	912	1/1200
61	-244	C_6	841	3/4096
62	-248	C_8	841	1/1452
65	-260	$C_4 \times C_2$	961	3/4624
66	-264	$C_4 \times C_2$	1056	3/4900
69	-276	$C_4 \times C_2$	1104	1/1728
70	-280	$C_2 \times C_2$	1089	3/5476
73	-292	C_4	1225	3/5776
74	-296	C_{10}	1225	1/2028
77	-308	$C_4 \times C_2$	1369	3/6400
78	-312	$C_2 \times C_2$	1369	3/6724
82	-328	C_4	1521	3/7396
85	-340	$C_2 \times C_2$	1681	135/559504
86	-344	C_{10}	1681	1/2700
89	-356	C_{12}	1849	3/8464
93	-372	$C_2 \times C_2$	2025	1/3072
94	-376	C_8	2025	3/9604

Table 3.4: Some fields $K = \mathbb{Q}(\sqrt{-d})$ with $h_K > 2$ and $|D_K| > 100$

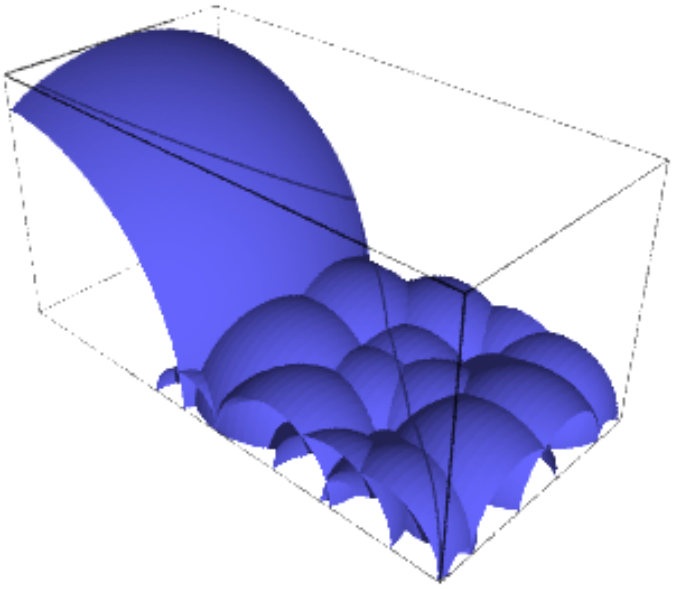

Figure 3.1: Bianchi diagram for $K = \mathbb{Q}(\sqrt{-39})$.

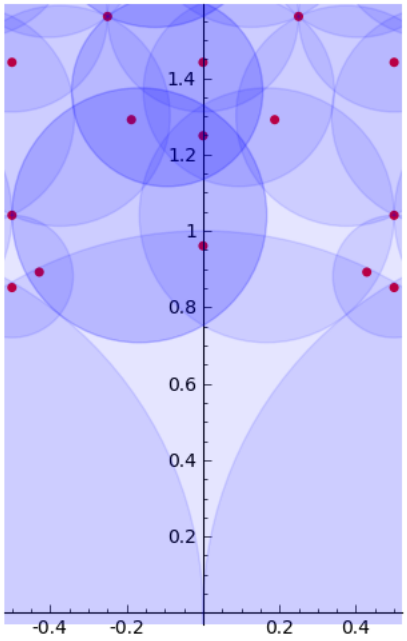

Figure 3.2: Projection on $\mathbb{C}$ of the Bianchi diagram for $K = \mathbb{Q}(\sqrt{-39})$.

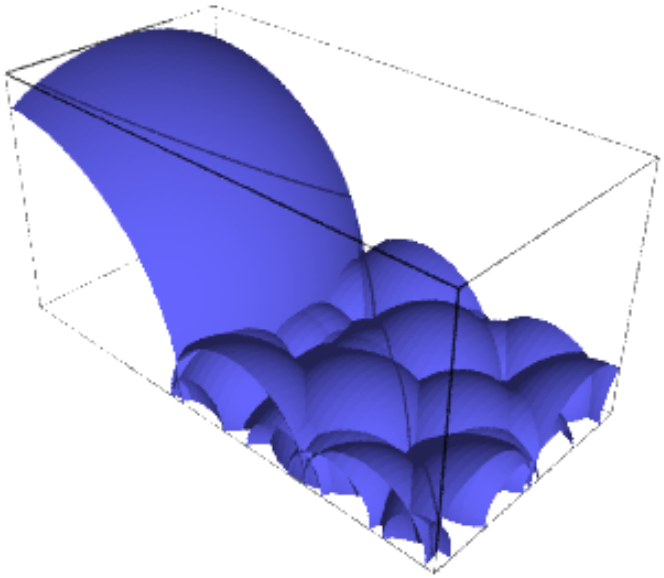

Figure 3.3: Bianchi diagram for $K = \mathbb{Q}(\sqrt{-47})$.

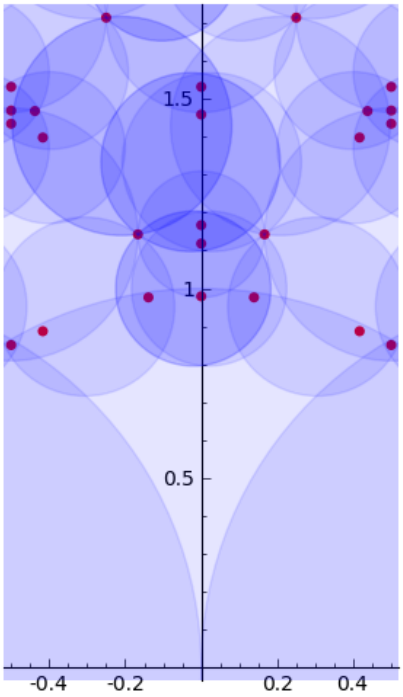

Figure 3.4: Projection on $\mathbb{C}$ of the Bianchi diagram for $K = \mathbb{Q}(\sqrt{-47})$.

3.4 Pseudo-Euclidean algorithms

Once we have determined the Bianchi diagram for Γ (the set of hemispheres that define the floor of $\mathcal{F}_K$) we can generalise the geometrical interpretation of the Euclidean algorithm that we gave at the beginning of §3.2.2 (page 65).

Throughout this section, let $S = \{S_{\alpha_1}, \ldots, S_{\alpha_m}\}$ be the list of hemispheres that form the floor of $\mathcal{F}_K$, and $\mathcal{S}_K = \{s_1, \ldots, s_r\}$ a list of the singular points (if any) lying in $\bar{F}_K$. For a given cusp $\beta \in \mathbb{P}^1(K)$, our pseudo-Euclidean algorithm should return a matrix M in Γ which sends β to a cusp with minimum denominator. From Corollary 3.2.16 we know that each singular cusp is a point with minimal denominator in its Γ-orbit. It is clear then that this matrix M will send principal cusps to ∞ and other cusps to the corresponding singular points.

To reduce the "size of the denominator" in our algorithm, we will use the matrices M_α that we described in Lemma 3.2.14. These matrices play the same role as the inversion matrix in Algorithm 3.2.2.

Algorithm 3.4.1 (Pseudo-Euclidean algorithm)**.** *Given a cusp $\beta \in \mathbb{P}^1(K)$, the algorithm finds a matrix M in Γ which sends β to the cusp ∞ or the corresponding singular point in its Γ-orbit.*

1. Initialize $M = Id$.
2. If $\beta = \infty$, we are done. Otherwise:
   (a) Find $M' \in \Gamma_\infty$ such that $M' \cdot \beta \in \bar{F}_K$.
   (b) Set $\beta = M' \cdot \beta$ and $M = M'M$.
3. If $\beta \in \mathcal{S}_K$, we are done. Otherwise:
   (a) Find a hemisphere S_{α_i} such that β lies inside C_{α_i}, and let M_{α_i} be the corresponding inversion matrix.
   (b) Set $\beta = M_{\alpha_i} \cdot \beta$ and $M = M_{\alpha_i} M$.
   (c) Go to step 2.
4. Return M.

Clearly for steps 2(a) and 3(a) in Algorithm 3.4.1 we need to know:

(i) a sub-algorithm which given $z \in \mathbb{C}$ returns a matrix $M' \in \Gamma_\infty$ such that $M' \cdot z \in \bar{F}_K$. Clearly M' can be expressed as a word in powers of T and U (recall §3.2.1);

(ii) a sub-algorithm which given a point $(z,t) \in \mathfrak{H}_3 \cup K$ finds the highest hemisphere S_{α_i} that covers (z,t);

(iii) a list of "inversion matrices", $\{M_{\alpha_1}, \ldots, M_{\alpha_m}\}$, one for each hemisphere S_{α_i} that is part of the floor of $\mathcal{B}_K$.

The sub-algorithm (i) is not difficult to program. Part (ii) is slightly more general than we need for Algorithm 3.4.1, but very easy to write as well. The inversion matrices in (iii) can be explicitly constructed as follows. Let $\alpha_i = \lambda/\mu$. We can choose λ, μ so that $\langle \lambda, \mu \rangle = \langle 1 \rangle = R$. Then there exist λ', $\mu' \in R$ so that $\lambda\lambda' + \mu\mu' = 1$, and we can define:

$$M = \begin{pmatrix} -\lambda' & -\mu' \\ \mu & -\lambda \end{pmatrix}.$$

Clearly $M \in \Gamma$, and $M \cdot \alpha_i = \infty$. Observe that this matrix is not unique; it may be replaced by AM for any A fixing ∞.

The same ingredients can be used to write an algorithm which will send interior points $(z,t) \in \mathfrak{H}_3$ to points in $\overline{\mathcal{F}_K}$. We need only to recall the method we described in Remark 3.2.13:

Algorithm 3.4.2 (Algorithm for interior points)**.** *Given a point $(z,t) \in \mathfrak{H}_3$, the algorithm returns a matrix M in Γ such that $M \cdot (z,t) \in \overline{\mathcal{F}_K}$.*

1. Initialize $M = Id$.
2. If $z \notin \bar{F}_K$:
   (a) Find $M' \in \Gamma_\infty$ such that $M' \cdot z \in \bar{F}_K$.
   (b) Set $z = M' \cdot z$ and $M = M'M$.
3. Find the hemisphere S_{α_i} that covers (z,t) with maximum height. If there is no such S_{α_i} (no hemisphere covers the point) we are done. Otherwise:
   (a) Let M_{α_i} be the corresponding inversion matrix for S_{α_i}.
   (b) Set $(z,t) = M_{\alpha_i} \cdot (z,t)$ and $M = M_{\alpha_i} M$.
   (c) Go to step 2.
4. Return M.

Remark 3.4.3. For homology computations based on modular symbols it is necessary to have a convenient way of writing down generators and relations for the homology group. This is done by transforming modular symbols into M-symbols and vice-versa. Conversion from M-symbols to modular symbols is practically trivial, and the pseudo-Euclidean algorithm gives a method to convert a modular symbol into a $\mathbb{Z}$-linear combination of M-symbols (a straightforward description of the method can be found in [22, pages 76-77]).

3.5 Simplifying the geometry with the group Δ

When the class group Cl_K of the field K has exponent 2 we can simplify the geometry by considering the fundamental domain for the action of the normaliser group Δ (which we introduced in §2.1) on the hyperbolic space. There is a theory of hemispheres for Δ analogous to the one we described for Γ in the previous sections. This theory was developed by J. Bygott in this thesis [6], where he applied it to the case $K = \mathbb{Q}(\sqrt{-5})$. The advantage of working with Δ is that we reduce the number of singular points, eliminating them when Cl_K is an elementary abelian 2-group. We have implemented the resulting algorithms (see some results in §3.5.1 below) so that we are able to obtain a fundamental domain for the action of Δ on $\mathfrak{H}_3$ when Cl_K has exponent 2. We now review briefly the main results of the theory, following the work in Bygott's thesis [6].

Recall from §2.1.1 that a cusp α is semi-principal (Definition 2.1.12) if and only if $M \cdot \alpha = \infty$ for some $M \in \Delta$. When $Cl_K = Cl_K[2]$ (with the notation of page 40) all cusps are Δ-equivalent.

In §3.2.2 we gave a general definition for a hemisphere attached to a cusp $\alpha \in K$:

$$S_\alpha = \left\{ (z,t) \in \mathfrak{H}_3 : |z-\alpha|^2 + t^2 = \frac{1}{\psi(\alpha)} \right\}.$$

For this general type of hemispheres we have an analogue to Lemma 3.2.8 (which was only for principal hemispheres):

Lemma 3.5.1. *Let $(z,t) \in \mathfrak{H}_3$. The set of cusps α such that S_α covers (z,t) is finite.*

Proof. We follow the proof in [6, Lemma 47]. Let $\alpha = \lambda/\mu$ be a cusp such that S_α covers the point (z,t). We can choose λ, μ so that $\mathrm{N}(\langle \lambda, \mu \rangle) \leq B$, with B for instance the Minkowski bound ([17, Theorem 35]), which is $\frac{2}{\pi}\sqrt{|D_K|}$ for imaginary

quadratic fields. The following inequality holds:

$$|z-\alpha|^2+t^2<\frac{1}{\psi(\alpha)}. \tag{3.5.1}$$

From the definition of the norm of the denominator $\psi(\alpha)$ and our choice of λ, μ, it follows that

$$\frac{\mathrm{N}(\mu)}{B}\leq\psi(\alpha),$$

which combined with (3.5.1) gives an upper bound for $|\mu|$:

$$|\mu|^2<\frac{B}{t^2}.$$

In particular, since $\mu\in R$, we know that there are only finitely many elements μ satisfying this condition. From (3.5.1) we can deduce that each μ in this finite set satisfies:

$$|\mu z-\lambda|^2<B-|\mu|^2t^2.$$

Hence for each fixed value of μ there is only a finite number of possible choices for λ. □

As with Lemma 3.2.8 in §3.2.2, we can now use the proof of Lemma 3.5.1 to obtain an algorithm to compute the finite set of hemispheres covering a given point in $\mathfrak{H}_3$.

We also have analogues of Lemma 3.2.10 and Lemma 3.2.14 for Δ:

Lemma 3.5.2. *Let* $M_\alpha=\begin{pmatrix}a & b\\ c & d\end{pmatrix}\in\Delta$, $\alpha=-d/c$ *a (semi-principal) cusp and* $(z',t')=M_\alpha\cdot(z,t)$. *Then*

$$\frac{t}{t'}=\psi(\alpha)(|z-\alpha|^2+t^2).$$

Hence $t'>t$ *if and only if* (z,t) *lies under the hemisphere* S_α. *Similarly,* $t'=t$ *if and only if* $(z,t)\in S_\alpha$, *and* $t'<t$ *if and only if* (z,t) *lies outside* S_α.

Proof. [6, Lemma 49] □

Lemma 3.5.3. *Let* $M_\alpha = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \in \Delta$, $\alpha = -d/c$ *a (semi-principal) cusp and* $\beta \in \mathbb{P}^1(K)$ *an arbitrary cusp. Then*

$$\psi(M_\alpha \cdot \beta) = \begin{cases} \psi(\beta) & \text{if } \alpha = \infty, \\ \psi(\alpha) & \text{if } \beta = \infty, \\ \psi(\alpha)\psi(\beta)|\beta - \alpha|^2 & \text{otherwise.} \end{cases}$$

Hence $\psi(M_\alpha \cdot \beta) < \psi(\beta)$ *if and only if* β *lies inside the circle* C_α. *Similarly,* $\psi(M_\alpha \cdot \beta) = \psi(\beta)$ *if and only if* $\beta \in C_\alpha$, *and* $\psi(M_\alpha \cdot \beta) > \psi(\beta)$ *if and only if* β *lies outside* C_α.

Proof. [6, Lemma 50]. □

In view of these two results, it makes sense to call a hemisphere S_α and the corresponding circle C_α *semi-principal* when the cusp α is a semi-principal cusp. We now describe a fundamental domain for Δ in the same way we did for Γ.

Definition 3.5.4. Define $\mathcal{B}_{K,\Delta}$ as the set of points $(z,t) \in \mathfrak{H}_3$ that lie above all principal and semi-principal hemispheres S_α.

The stabiliser Δ_∞ is given by ([6, Lemma 45]),

$$\Gamma_\infty = K^\times \cdot \left\{ \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix} : a \in R^\times, b \in R \right\}.$$

Hence the region F_K that we described in §3.2.1 is also a fundamental region for the complex plane $\mathbb{C}$ with respect to the action of Δ_∞. In particular:

Theorem 3.5.5. *Let* F_K *be a fundamental region for* $\mathbb{C}$ *as described above in* (3.2.2) *or* (3.2.3). *Then the set*

$$\mathcal{F}_{K,\Delta} = \{(z,t) \in \mathcal{B}_{K,\Delta} : z \in F_K\}$$

is a fundamental region for the action of Δ *on* $\mathfrak{H}_3$.

Proof. [6, Theorem 52]. □

Theorem 3.5.6. *The boundary of* $\mathcal{F}_{K,\Delta}$ *is defined by finitely many geodesic surfaces.*

Proof. [6, Theorem 53]. □

Again, points that lie inside no semi-principal C_α are special. We have then a theory of singular points, exactly as we did for Γ.

Definition 3.5.7. A point $z \in \mathbb{C}$ is Δ-*singular* if it lies inside no principal or semi-principal circle C_α.

Corollary 3.5.8. *Let $\beta \in K$. Then the cusp β is Δ-singular if and only if $\psi(\beta)$ is minimal for points in the Δ-orbit of β.*

Proof. See [6, Corollary 51]. □

Clearly if a cusp $\beta \in K$ is Δ-singular, it is singular. It follows at once from Corollary 3.5.8 that there are no Δ-singular points when $Cl_K = Cl_K[2]$. Consequently, for any imaginary quadratic field K whose class group has exponent 2 we can determine a fundamental domain for Δ as easily as we computed fundamental domains respect to Γ for class number one fields.

Remark 3.5.9. Our interest in the fundamental domain $\mathcal{F}_K$ for Γ lies in the fact that its geometry will provide the information necessary to compute the rational 1-homology of $\Gamma_0(\mathfrak{n})\backslash\mathfrak{H}_3^*$. From the tessellation of $\mathfrak{H}_3$ obtained from the fundamental domain one can read off generators and relations for the homology group. This information is described in terms of M-symbols of level $\mathfrak{n}$. Since the M-symbols of level $\mathfrak{n}$ provide both a set of representatives for $\Gamma_0(\mathfrak{n})$ in Γ and a set of representatives for $\Delta_0(\mathfrak{n})$ in Δ (recall Proposition 2.1.16), we can work with $\Delta_0(\mathfrak{n})\backslash\mathfrak{H}_3^*$ instead, and read our geometrical information from the simpler fundamental domain $\mathcal{F}_{K,\Delta}$. This is explained with more detail in [6, pages 74-76].

3.5.1 Examples

Without much effort it was possible to modify our code so that it computes the fundamental region with respect to Δ-action for fields whose class group has exponent 2. In these cases we have no Δ-singular points and the computations are as easy as for class number one fields. What is more, the geometry of the fundamental region becomes simpler.

In the following table we show data for some fundamental regions $\mathcal{F}_{K,\Delta}$, which we have computed for a few fields K with $Cl_K \cong C_2$ or $Cl_K \cong C_2 \times C_2$. As in §3.3, we include information about the lowest vertex of the tessellation ("min t^2") and a bound for the size of the hemispheres that form the Bianchi diagram, which in this case is given by the maximum value of $\psi(\alpha)$ for all $S_\alpha \in \mathcal{B}_{K,\Delta}$ (recall that for $\alpha = \lambda/\mu$ principal, $\psi(\alpha) = |\mu|^2$). We can compare the results below with the corresponding numbers obtained for these same fields when we computed $\mathcal{F}_K$ (see tables in §3.3).

d	D_K	Cl_K	max $\psi(\alpha)$	min t^2
5	-20	C_2	2	1/5
6	-24	C_2	2	1/12
10	-40	C_2	7	13/180
13	-52	C_2	9	3/52
15	-15	C_2	2	1/3
21	-84	C_2	16	1/28
22	-88	C_2	19	1/44
30	-120	$C_2 \times C_2$	25	1/25
33	-132	$C_2 \times C_2$	11	1/66
35	-35	C_2	4	1/7
37	-148	C_2	31	3/148
42	-168	$C_2 \times C_2$	25	5/168
51	-51	C_2	9	2/51
57	-228	$C_2 \times C_2$	49	1/114
58	-232	C_2	64	69/5684
70	-280	$C_2 \times C_2$	49	3/196
78	-312	$C_2 \times C_2$	79	1/108
91	-91	C_2	9	2/63

Table 3.5: Fields $K = \mathbb{Q}(\sqrt{-d})$ with $Cl_K = Cl_K[2]$

3.6 Tessellations and homology

From a fundamental domain like the ones described in §3.2 or §3.5, it is possible to find a tessellation of $\mathfrak{H}_3$ by "ideal" polyhedra, that is, hyperbolic polyhedra all of whose vertices are at cusps. The edges, face relations and other geometrical information from the polyhedra would give us the information needed to compute the rational 1-homology group of a quotient $G\backslash\mathfrak{H}_3^*$, with G a subgroup of finite index in Γ (e.g. $G = \Gamma_0(\mathfrak{n})$).

To obtain a tessellation we need only to cut our fundamental region $\mathcal{F}_K$ (or $\mathcal{F}_{K,\Delta}$) into pieces, and then glue together translates of these pieces to form the hyperbolic polyhedra that tessellate $\mathfrak{H}_3$. These techniques were used in the theses of E. Whitley [35], J. Bygott [6] and M. Lingham [22]. In principle their methods could be generalised, but due to the differences in the geometry for each number field all explicit computations were done for specific cases (non Euclidean imaginary quadratic fields with class number one in [35], $K = \mathbb{Q}(\sqrt{-5})$ in [6], and $K = \mathbb{Q}(\sqrt{-d})$ with $d = 23, 31$ in [22]).

In [22] M. Lingham outlines a general method to obtain a tessellation from a fundamental domain. When there are no singular points, the algorithms he describes [22, pages 56-58] are valid and their implementation straightforward. We can apply this construction then to imaginary quadratic fields K with class number $h_K = 1$ or, thanks to the introduction of the normaliser Δ, class group Cl_K of exponent two. The strategy he uses to deal with singular points for the fields $K = \mathbb{Q}(\sqrt{-23})$ and $K = \mathbb{Q}(\sqrt{-31})$ will work for any other case, but it depends on the shape of the fundamental domain around the singular point. At the time of submitting this thesis we have not yet found a way to implement a general algorithm.

Other techniques

Recent work by Paul Gunnells and Dan Yasaki [19, 36] provides a new method to find the hyperbolic polyhedra that tessellate $\mathfrak{H}_3$. Their techniques are not based on the geometry of the field, so they can be more easily generalized.

A rough outline of this method is as follows. For $K/\mathbb{Q}$ a number field, the space of positive definite binary Hermitian forms over K forms an open cone in a real vector space. There is a natural decomposition of this cone into polyhedral cones corresponding to the facets of the Voronoï polyhedron [19]. If K is an imaginary quadratic field, the top-dimensional polyhedral cones of the decomposition correspond to perfect forms and descend to ideal polytopes in $\mathfrak{H}_3$. These polytopes form a tessellation analogous to the one we would obtain using the geometrical approach

we described above, which was based on knowledge of a fundamental region for $\mathfrak{H}_3$.

In [36] Yasaki gives a method to compute an initial perfect form, which is the necessary input for a general algorithm by Gunnells [19] that classifies perfect forms under $\mathrm{GL}(2, R)$ action. Dan Yasaki has developed a Magma [5] package which in principle can find a hyperbolic tessellation for any imaginary quadratic field. Results for a number of fields are listed in his paper [36].

Other techniques can be used to compute rational homology of Bianchi groups which do not involve a description of a fundamental domain for $\mathfrak{H}_3$. Vogtmann [34] computed rational (and integral [28]) homology by determining the cell structure and homology of a 2-cellular retract of $\mathfrak{H}_3$ which is invariant under the action of $\mathrm{SL}(2, R)$. Some code was written (unfortunately it is lost) and the calculations for K with discriminant $D_K > -100$ can be found in [34]. The techniques described by Vogtmann work for any imaginary quadratic field, but they are not very convenient for homology computations at higher levels.

Remark 3.6.1. It has just come to our attention that very recently another Ph.D. student had implemented Swan's methods to obtain fundamental domains for Bianchi groups. In his thesis, A. Rham [24] reviews Swan's paper [32] and describes his own implementation of the methods in Pari/GP [33], which he has applied to imaginary quadratic fields of class number 1 and 2. Using an approach similar to the one taken by Schwermer and Vogtman [28], Rham considers a 2-cellular retract for $\mathfrak{H}_3$ which in his case preserves the geometry of the fundamental domain $\mathcal{F}_K$. With this data he is able to compute integral homology of Bianchi groups [24, 25].

Appendix A

Implementation

A.1 Implementation of the algorithms in Chapter 1

The source code for the implementation of the algorithms in Chapter 1 is now part of the official *Sage* release. The corresponding files are available at
`http://hg.sagemath.org/sage-main/file/4.6/sage/modular/cusps_nf.py`
`http://hg.sagemath.org/sage-main/file/4.6/sage/modular/modsym/p1list_nf.py`

A complete description of the implementation can be found in the *Sage's Reference Manual* (`http://www.sagemath.org/doc/reference/`). Below we reproduce the relevant sections of the Manual:

1 Section 52.14. List of Manin symbols over number fields.

2 Section 55.10. The set $\mathbb{P}^1(K)$ of cusps of a number field K.

```
sage: v = [((int(0),3), (int(1),-1)), ((int(1),1), (int(3),1)), ((int(2),1),(int(3)
sage: rels = set(v)
sage: n = 6
sage: from sage.modular.modsym.relation_matrix import sparse_2term_quotient
sage: sparse_2term_quotient(rels, n, QQ)
[(3, -1/3), (3, -1), (3, -1), (3, 1), (5, 1), (5, 1)]
```

52.14 Lists of Manin symbols (elements of $\mathbb{P}^1(R/N)$) over number fields.

Lists of elements of $\mathbb{P}^1(R/N)$ where R is the ring of integers of a number field K and N is an integral ideal.

AUTHORS:

- Maite Aranes (2009): Initial version

EXAMPLES:

We define a P1NFList:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a^2 - a + 1)
sage: P = P1NFList(N); P
The projective line over the ring of integers modulo the Fractional ideal (5, a^2 - a +
```

List operations with the P1NFList:

```
sage: len(P)
26
sage: [p for p in P]
[M-symbol (0: 1) of level Fractional ideal (5, a^2 - a + 1),
...
M-symbol (1: 2*a^2 + 2*a) of level Fractional ideal (5, a^2 - a + 1)]
```

The elements of the P1NFList are M-symbols:

```
sage: type(P[2])
<class 'sage.modular.modsym.p1list_nf.MSymbol'>
```

Definition of MSymbols:

```
sage: alpha = MSymbol(N, 3, a^2); alpha
M-symbol (3: a^2) of level Fractional ideal (5, a^2 - a + 1)
```

Find the index of the class of an M-Symbol $(c : d)$ in the list:

```
sage: i = P.index(alpha)
sage: P[i].c*alpha.d - P[i].d*alpha.c in N
True
```

Lift an MSymbol to a matrix in $SL(2, R)$:

```
sage: alpha = MSymbol(N, a + 2, 3*a^2)
sage: alpha.lift_to_sl2_Ok()
[1, -4*a^2 + 9*a - 21, a + 2, a^2 - 3*a + 3]
sage: Ok = k.ring_of_integers()
sage: M = Matrix(Ok, 2, alpha.lift_to_sl2_Ok())
sage: det(M)
1
sage: M[1][1] - alpha.d in N
True
```

Lift an MSymbol from P1NFList to a matrix in $SL(2,R)$

```
sage: P[3]
M-symbol (1: -2*a) of level Fractional ideal (5, a^2 - a + 1)
sage: P.lift_to_sl2_Ok(3)
[0, -1, 1, -2*a]
```

class MSymbol (*N, c, d=None, check=True*)

Bases: sage.structure.sage_object.SageObject

The constructor for an M-symbol over a number field.

INPUT:

- N – integral ideal (the modulus or level).
- c – integral element of the underlying number field or an MSymbol of level N.
- d – (optional) when present, it must be an integral element such that <c> + <d> + N = R, where R is the corresponding ring of integers.
- check – bool (default True). If check=False the constructor does not check the condition <c> + <d> + N = R.

OUTPUT:

An M-symbol modulo the given ideal N, i.e. an element of the projective line $\mathbb{P}^1(R/N)$, where R is the ring of integers of the underlying number field.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(a + 1, 2)
sage: MSymbol(N, 3, a^2 + 1)
M-symbol (3: a^2 + 1) of level Fractional ideal (2, a + 1)
```

We can give a tuple as input:

```
sage: MSymbol(N, (1, 0))
M-symbol (1: 0) of level Fractional ideal (2, a + 1)
```

We get an error if <c>, <d> and N are not coprime:

```
sage: MSymbol(N, 2*a, a - 1)
...
ValueError: (2*a, a - 1) is not an element of P1(R/N).
sage: MSymbol(N, (0, 0))
...
ValueError: (0, 0) is not an element of P1(R/N).
```

Saving and loading works:

```
sage: alpha = MSymbol(N, 3, a^2 + 1)
sage: loads(dumps(alpha))==alpha
True
```

N()

Returns the level or modulus of this MSymbol.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(3, a - 1)
sage: alpha = MSymbol(N, 3, a)
sage: alpha.N()
Fractional ideal (3, 1/2*a - 1/2)
```

c

Returns the first coefficient of the M-symbol.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(a + 1, 2)
sage: alpha = MSymbol(N, 3, a^2 + 1)
sage: alpha.c # indirect doctest
3
```

d

Returns the second coefficient of the M-symbol.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(a + 1, 2)
sage: alpha = MSymbol(N, 3, a^2 + 1)
sage: alpha.d # indirect doctest
a^2 + 1
```

lift_to_sl2_Ok()

Lift the MSymbol to an element of $SL(2, Ok)$, where Ok is the ring of integers of the corresponding number field.

OUTPUT:

A list of integral elements $[a, b, c', d']$ that are the entries of a 2x2 matrix with determinant 1. The lower two entries are congruent (modulo the level) to the coefficients c, d of the MSymbol self.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(3, a - 1)
sage: alpha = MSymbol(N, 3*a + 1, a)
sage: alpha.lift_to_sl2_Ok()
[0, -1, 1, a]
```

normalize(*with_scalar=False*)

Returns a normalized MSymbol (a canonical representative of an element of $\mathbb{P}^1(R/N)$) equivalent to `self`.

INPUT:

- with_scalar – bool (default False)

OUTPUT:

- (only if with_scalar=True) a transforming scalar u, such that $(u*c', u*d')$ is congruent to $(c:d)$ (mod N), where $(c:d)$ are the coefficients of self and N is the level.
- a normalized MSymbol (c': d') equivalent to self.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(3, a - 1)
sage: alpha1 = MSymbol(N, 3, a); alpha1
M-symbol (3: a) of level Fractional ideal (3, 1/2*a - 1/2)
sage: alpha1.normalize()
M-symbol (0: 1) of level Fractional ideal (3, 1/2*a - 1/2)
sage: alpha2 = MSymbol(N, 4, a + 1)
sage: alpha2.normalize()
M-symbol (1: -a) of level Fractional ideal (3, 1/2*a - 1/2)
```

We get the scaling factor by setting with_scalar=True:

```
sage: alpha1.normalize(with_scalar=True)
(a, M-symbol (0: 1) of level Fractional ideal (3, 1/2*a - 1/2))
sage: r, beta1 = alpha1.normalize(with_scalar=True)
sage: r*beta1.c - alpha1.c in N
True
sage: r*beta1.d - alpha1.d in N
True
sage: r, beta2 = alpha2.normalize(with_scalar=True)
sage: r*beta2.c - alpha2.c in N
True
sage: r*beta2.d - alpha2.d in N
True
```

tuple()

Returns the MSymbol as a list (c, d).

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(3, a - 1)
sage: alpha = MSymbol(N, 3, a); alpha
M-symbol (3: a) of level Fractional ideal (3, 1/2*a - 1/2)
sage: alpha.tuple()
(3, a)
```

class P1NFList(N)

Bases: sage.structure.sage_object.SageObject

The class for $\mathbb{P}^1(R/N)$, the projective line modulo N, where R is the ring of integers of a number field K and N is an integral ideal.

INPUT:

- N - integral ideal (the modulus or level).

OUTPUT:

A P1NFList object representing $\mathbb{P}^1(R/N)$.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a + 1)
sage: P = P1NFList(N); P
The projective line over the ring of integers modulo the Fractional ideal (5, a + 1
```

Saving and loading works.

```
sage: loads(dumps(P)) == P
True
```

N()

Returns the level or modulus of this P1NFList.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 31)
sage: N = k.ideal(5, a + 3)
sage: P = P1NFList(N)
sage: P.N()
Fractional ideal (5, 1/2*a + 3/2)
```

apply_J_epsilon(*i*, *e1*, *e2=1*)

Applies the matrix J_ϵ = [e1, 0, 0, e2] to the i-th M-Symbol of the list.

e1, e2 are units of the underlying number field.

INPUT:

- `i` – integer
- `e1` – unit
- `e2` – unit (default 1)

OUTPUT:

integer – the index of the M-Symbol obtained by the right action of the matrix J_ϵ = [e1, 0, 0, e2] on the i-th M-Symbol.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a + 1)
sage: P = P1NFList(N)
sage: u = k.unit_group().gens(); u
[-1, 2*a^2 + 4*a - 1]
sage: P.apply_J_epsilon(4, -1)
2
sage: P.apply_J_epsilon(4, u[0], u[1])
1
```

```
sage: k.<a> = NumberField(x^4 + 13*x - 7)
sage: N = k.ideal(a + 1)
sage: P = P1NFList(N)
sage: u = k.unit_group().gens(); u
```

```
[-1, a^3 + a^2 + a + 12, a^3 + 3*a^2 - 1]
sage: P.apply_J_epsilon(3, u[2]^2)==P.apply_J_epsilon(P.apply_J_epsilon(3, u[2]
True
```

apply_S(*i*)

Applies the matrix S = [0, -1, 1, 0] to the i-th M-Symbol of the list.

INPUT:

- i – integer

OUTPUT:

integer – the index of the M-Symbol obtained by the right action of the matrix S = [0, -1, 1, 0] on the i-th M-Symbol.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a + 1)
sage: P = P1NFList(N)
sage: j = P.apply_S(P.index_of_normalized_pair(1, 0))
sage: P[j]
M-symbol (0: 1) of level Fractional ideal (5, a + 1)
```

We test that S has order 2:

```
sage: j = randint(0,len(P)-1)
sage: P.apply_S(P.apply_S(j))==j
True
```

apply_TS(*i*)

Applies the matrix TS = [1, -1, 0, 1] to the i-th M-Symbol of the list.

INPUT:

- i – integer

OUTPUT:

integer – the index of the M-Symbol obtained by the right action of the matrix TS = [1, -1, 0, 1] on the i-th M-Symbol.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a + 1)
sage: P = P1NFList(N)
sage: P.apply_TS(3)
2
```

We test that TS has order 3:

```
sage: j = randint(0,len(P)-1)
sage: P.apply_TS(P.apply_TS(P.apply_TS(j)))==j
True
```

apply_T_alpha(*i, alpha=1*)

Applies the matrix T_alpha = [1, alpha, 0, 1] to the i-th M-Symbol of the list.

INPUT:

- i – integer
- alpha – element of the corresponding ring of integers(default 1)

OUTPUT:

integer – the index of the M-Symbol obtained by the right action of the matrix T_alpha = [1, alpha, 0, 1] on the i-th M-Symbol.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a + 1)
sage: P = P1NFList(N)
sage: P.apply_T_alpha(4, a^ 2 - 2)
3
```

We test that T_a*T_b = T_(a+b):

```
sage: P.apply_T_alpha(3, a^2 - 2)==P.apply_T_alpha(P.apply_T_alpha(3,a^2),-2)
True
```

index(*c, d=None, with_scalar=False*)

Returns the index of the class of the pair (c, d) in the fixed list of representatives of $\mathbb{P}^1(R/N)$.

INPUT:

- c – integral element of the corresponding number field, or an MSymbol.
- d – (optional) when present, it must be an integral element of the number field such that (c, d) defines an M-symbol of level N.
- with_scalar – bool (default False)

OUTPUT:

- u - the normalizing scalar (only if with_scalar=True)
- i - the index of (c, d) in the list.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 31)
sage: N = k.ideal(5, a + 3)
sage: P = P1NFList(N)
sage: P.index(3,a)
5
sage: P[5]==MSymbol(N, 3, a).normalize()
True
```

We can give an MSymbol as input:

```
sage: alpha = MSymbol(N, 3, a)
sage: P.index(alpha)
5
```

We cannot look for the class of an MSymbol of a different level:

```
sage: M = k.ideal(a + 1)
sage: beta = MSymbol(M, 0, 1)
sage: P.index(beta)
```

```
...
ValueError: The MSymbol is of a different level
```

If we are interested in the transforming scalar:

```
sage: alpha = MSymbol(N, 3, a)
sage: P.index(alpha, with_scalar=True)
(-a, 5)
sage: u, i = P.index(alpha, with_scalar=True)
sage: (u*P[i].c - alpha.c in N) and (u*P[i].d - alpha.d in N)
True
```

index_of_normalized_pair(*c*, *d=None*)

Returns the index of the class (c, d) in the fixed list of representatives of $(P)^1(R/N)$.

INPUT:

- `c` – integral element of the corresponding number field, or a normalized MSymbol.
- `d` – (optional) when present, it must be an integral element of the number field such that (c, d) defines a normalized M-symbol of level N.

OUTPUT:

- `i` - the index of (c, d) in the list.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 31)
sage: N = k.ideal(5, a + 3)
sage: P = P1NFList(N)
sage: P.index_of_normalized_pair(1, 0)
3
sage: j = randint(0,len(P)-1)
sage: P.index_of_normalized_pair(P[j])==j
True
```

lift_to_sl2_Ok(*i*)

Lift the i-th element of this P1NFList to an element of $SL(2, R)$, where R is the ring of integers of the corresponding number field.

INPUT:

- `i` - integer (index of the element to lift)

OUTPUT:

If the i-th element is $(c : d)$, the function returns a list of integral elements $[a, b, c', d']$ that defines a 2x2 matrix with determinant 1 and such that $c = c' \pmod N$ and $d = d' \pmod N$.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(3)
sage: P = P1NFList(N)
sage: len(P)
16
sage: P[5]
M-symbol (1/2*a + 1/2: -a) of level Fractional ideal (3)
sage: P.lift_to_sl2_Ok(5)
[1, -2, 1/2*a + 1/2, -a]
```

```
sage: Ok = k.ring_of_integers()
sage: L = [Matrix(Ok, 2, P.lift_to_sl2_Ok(i)) for i in range(len(P))]
sage: all([det(L[i]) == 1 for i in range(len(L))])
True
```

list()

Returns the underlying list of this P1NFList object.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a+1)
sage: P = P1NFList(N)
sage: type(P)
<class 'sage.modular.modsym.p1list_nf.P1NFList'>
sage: type(P.list())
<type 'list'>
```

normalize(*c*, *d=None*, *with_scalar=False*)

Returns a normalised element of $\mathbb{P}^1(R/N)$.

INPUT:

- c – integral element of the underlying number field, or an MSymbol.
- d – (optional) when present, it must be an integral element of the number field such that (c, d) defines an M-symbol of level N.
- with_scalar – bool (default False)

OUTPUT:

- (only if with_scalar=True) a transforming scalar u, such that $(u * c', u * d')$ is congruent to $(c : d)$ (mod N).
- a normalized MSymbol (c': d') equivalent to $(c : d)$.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 31)
sage: N = k.ideal(5, a + 3)
sage: P = P1NFList(N)
sage: P.normalize(3, a)
M-symbol (1: 2*a) of level Fractional ideal (5, 1/2*a + 3/2)
```

We can use an MSymbol as input:

```
sage: alpha = MSymbol(N, 3, a)
sage: P.normalize(alpha)
M-symbol (1: 2*a) of level Fractional ideal (5, 1/2*a + 3/2)
```

If we are interested in the normalizing scalar:

```
sage: P.normalize(alpha, with_scalar=True)
(-a, M-symbol (1: 2*a) of level Fractional ideal (5, 1/2*a + 3/2))
sage: r, beta = P.normalize(alpha, with_scalar=True)
sage: (r*beta.c - alpha.c in N) and (r*beta.d - alpha.d in N)
True
```

P1NFList_clear_level_cache()
Clear the global cache of data for the level ideals.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(a+1)
sage: alpha = MSymbol(N, 2*a^2, 5)
sage: alpha.normalize()
M-symbol (-4*a^2: 5*a^2) of level Fractional ideal (a + 1)
sage: sage.modular.modsym.p1list_nf._level_cache
{Fractional ideal (a + 1): (...)}
sage: sage.modular.modsym.p1list_nf.P1NFList_clear_level_cache()
sage: sage.modular.modsym.p1list_nf._level_cache
{}
```

lift_to_sl2_Ok(*N, c, d*)
Lift a pair (c, d) to an element of $SL(2, O_k)$, where O_k is the ring of integers of the corresponding number field.

INPUT:

- N – number field ideal
- c – integral element of the number field
- d – integral element of the number field

OUTPUT:

A list [a, b, c', d'] of integral elements that are the entries of a 2x2 matrix with determinant 1. The lower two entries are congruent to c, d modulo the ideal N.

EXAMPLES:

```
sage: from sage.modular.modsym.p1list_nf import lift_to_sl2_Ok
sage: k.<a> = NumberField(x^2 + 23)
sage: Ok = k.ring_of_integers(k)
sage: N = k.ideal(3)
sage: M = Matrix(Ok, 2, lift_to_sl2_Ok(N, 1, a))
sage: det(M)
1
sage: M = Matrix(Ok, 2, lift_to_sl2_Ok(N, 0, a))
sage: det(M)
1
sage: (M[1][0] in N) and (M[1][1] - a in N)
True
sage: M = Matrix(Ok, 2, lift_to_sl2_Ok(N, 0, 0))
...
ValueError: Cannot lift (0, 0) to an element of Sl2(Ok).


sage: k.<a> = NumberField(x^3 + 11)
sage: Ok = k.ring_of_integers(k)
sage: N = k.ideal(3, a - 1)
sage: M = Matrix(Ok, 2, lift_to_sl2_Ok(N, 2*a, 0))
sage: det(M)
1
sage: (M[1][0] - 2*a in N) and (M[1][1] in N)
True
sage: M = Matrix(Ok, 2, lift_to_sl2_Ok(N, 4*a^2, a + 1))
sage: det(M)
```

```
1
sage: (M[1][0] - 4*a^2 in N) and (M[1][1] - (a+1) in N)
True

sage: k.<a> = NumberField(x^4 - x^3 -21*x^2 + 17*x + 133)
sage: Ok = k.ring_of_integers(k)
sage: N = k.ideal(7, a)
sage: M = Matrix(Ok, 2, lift_to_sl2_Ok(N, 0, a^2 - 1))
sage: det(M)
1
sage: (M[1][0] in N) and (M[1][1] - (a^2-1) in N)
True
sage: M = Matrix(Ok, 2, lift_to_sl2_Ok(N, 0, 7))
...
ValueError: <0> + <7> and the Fractional ideal (7, a) are not coprime.
```

make_coprime (*N, c, d*)

Returns (c, d') so d' is congruent to d modulo N, and such that c and d' are coprime (<c> + <d'> = R).

INPUT:

- N – number field ideal
- c – integral element of the number field
- d – integral element of the number field

OUTPUT:

A pair (c, d') where c, d' are integral elements of the corresponding number field, with d' congruent to d mod N, and such that <c> + <d'> = R (R being the corresponding ring of integers).

EXAMPLES:

```
sage: from sage.modular.modsym.p1list_nf import make_coprime
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(3, a - 1)
sage: c = 2*a; d = a + 1
sage: N.is_coprime(k.ideal(c, d))
True
sage: k.ideal(c).is_coprime(d)
False
sage: c, dp = make_coprime(N, c, d)
sage: k.ideal(c).is_coprime(dp)
True
```

p1NFlist (*N*)

Returns a list of the normalized elements of $\mathbb{P}^1(R/N)$, where N is an integral ideal.

INPUT:

- N - integral ideal (the level or modulus).

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(3)
sage: from sage.modular.modsym.p1list_nf import p1NFlist, psi
sage: len(p1NFlist(N))==psi(N)
True
```

psi(N)

The index $[\Gamma : \Gamma_0(N)]$, where $\Gamma = GL(2, R)$ for R the corresponding ring of integers, and $\Gamma_0(N)$ standard congruence subgroup.

EXAMPLES:

```
sage: from sage.modular.modsym.p1list_nf import psi
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(3, a - 1)
sage: psi(N)
4
```

```
sage: k.<a> = NumberField(x^2 + 23)
sage: N = k.ideal(5)
sage: psi(N)
26
```

55.10 The set $\mathbb{P}^1(K)$ of cusps of a number field K.

AUTHORS:

- Maite Aranes (2009): Initial version

EXAMPLES:

The space of cusps over a number field k:

```
sage: k.<a> = NumberField(x^2 + 5)
sage: kCusps = NFCusps(k); kCusps
Set of all cusps of Number Field in a with defining polynomial x^2 + 5
sage: kCusps is NFCusps(k)
True
```

Define a cusp over a number field:

```
sage: NFCusp(k, a, 2/(a+1))
Cusp [a - 5: 2] of Number Field in a with defining polynomial x^2 + 5
sage: kCusps((a,2))
Cusp [a: 2] of Number Field in a with defining polynomial x^2 + 5
sage: NFCusp(k,oo)
Cusp Infinity of Number Field in a with defining polynomial x^2 + 5
```

Different operations with cusps over a number field:

```
sage: alpha = NFCusp(k, 3, 1/a + 2); alpha
Cusp [a + 10: 7] of Number Field in a with defining polynomial x^2 + 5
sage: alpha.numerator()
a + 10
sage: alpha.denominator()
7
sage: alpha.ideal()
Fractional ideal (7, a + 3)
sage: alpha.ABmatrix()
[a + 10, 3*a - 1, 7, 2*a]
sage: alpha.apply([0, 1, -1,0])
Cusp [7: -a - 10] of Number Field in a with defining polynomial x^2 + 5
```

Check Gamma0(N)-equivalence of cusps:

```
sage: N = k.ideal(3)
sage: alpha = NFCusp(k, 3, a + 1)
sage: beta = kCusps((2, a - 3))
sage: alpha.is_Gamma0_equivalent(beta, N)
True
```

Obtain transformation matrix for equivalent cusps:

```
sage: t, M = alpha.is_Gamma0_equivalent(beta, N, Transformation=True)
sage: M
[-2*a + 4, 3*a + 4, 5*a - 2, -3*a - 13]
sage: alpha.apply(M) == beta
True
```

List representatives for Gamma_0(N) - equivalence classes of cusps:

```
sage: Gamma0_NFCusps(N)
[Cusp [0: 1] of Number Field in a with defining polynomial x^2 + 5,
Cusp [1: 3] of Number Field in a with defining polynomial x^2 + 5,
...]
```

Gamma0_NFCusps(N)

Returns a list of inequivalent cusps for $\Gamma_0(N)$, i.e., a set of representatives for the orbits of `self` on $\mathbb{P}^1(k)$.

INPUT:

- `N` – an integral ideal of the number field k (the level).

OUTPUT:

A list of inequivalent number field cusps.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 5)
sage: N = k.ideal(3)
sage: L = Gamma0_NFCusps(N)
```

The cusps in the list are inequivalent:

```
sage: all([not L[i].is_Gamma0_equivalent(L[j], N) for i, j in \
                                        mrange([len(L), len(L)]) if i<j])
True
```

We test that we obtain the right number of orbits:

```
sage: from sage.modular.cusps_nf import number_of_Gamma0_NFCusps
sage: len(L) == number_of_Gamma0_NFCusps(N)
True
```

Another example:

```
sage: k.<a> = NumberField(x^4 - x^3 -21*x^2 + 17*x + 133)
sage: N = k.ideal(5)
sage: from sage.modular.cusps_nf import number_of_Gamma0_NFCusps
sage: len(Gamma0_NFCusps(N)) == number_of_Gamma0_NFCusps(N) # long time (over 1 sec
True
```

class NFCusp(*number_field, a, b=None, parent=None, lreps=None*)

Bases: `sage.structure.element.Element`

Creates a number field cusp, i.e., an element of $\mathbb{P}^1(k)$.

A cusp on a number field is either an element of the field or infinity, i.e., an element of the projective line over the number field. It is stored as a pair (a,b), where a, b are integral elements of the number field.

INPUT:

- `number_field` – the number field over which the cusp is defined.
- `a` – it can be a number field element (integral or not), or a number field cusp.
- `b` – (optional) when present, it must be either Infinity or coercible to an element of the number field.
- `lreps` – (optional) a list of chosen representatives for all the ideal classes of the field. When given, the representative of the cusp will be changed so its associated ideal is one of the ideals in the list.

OUTPUT:

`[a: b]` – a number field cusp.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 5)
sage: NFCusp(k, a, 2)
Cusp [a: 2] of Number Field in a with defining polynomial x^2 + 5
sage: NFCusp(k, (a,2))
Cusp [a: 2] of Number Field in a with defining polynomial x^2 + 5
sage: NFCusp(k, a, 2/(a+1))
Cusp [a - 5: 2] of Number Field in a with defining polynomial x^2 + 5
```

Cusp Infinity:

```
sage: NFCusp(k, 0)
Cusp [0: 1] of Number Field in a with defining polynomial x^2 + 5
sage: NFCusp(k, oo)
Cusp Infinity of Number Field in a with defining polynomial x^2 + 5
sage: NFCusp(k, 3*a, oo)
Cusp [0: 1] of Number Field in a with defining polynomial x^2 + 5
sage: NFCusp(k, a + 5, 0)
Cusp Infinity of Number Field in a with defining polynomial x^2 + 5
```

Saving and loading works:

```
sage: alpha = NFCusp(k, a, 2/(a+1))
sage: loads(dumps(alpha))==alpha
True
```

Some tests:

```
sage: I*I
-1
sage: NFCusp(k, I)
...
TypeError: Unable to convert I to a cusp of the number field
```

```
sage: NFCusp(k, oo, oo)
...
TypeError: Unable to convert (+Infinity, +Infinity) to a cusp of the number field
```

```
sage: NFCusp(k, 0, 0)
...
TypeError: Unable to convert (0, 0) to a cusp of the number field
```

```
sage: NFCusp(k, "a + 2", a)
Cusp [-2*a + 5: 5] of Number Field in a with defining polynomial x^2 + 5
```

```
sage: NFCusp(k, NFCusp(k, oo))
Cusp Infinity of Number Field in a with defining polynomial x^2 + 5
sage: c = NFCusp(k, 3, 2*a)
sage: NFCusp(k, c, a + 1)
Cusp [-a - 5: 20] of Number Field in a with defining polynomial x^2 + 5
sage: L.<b> = NumberField(x^2 + 2)
```

```
sage: NFCusp(L, c)
...
ValueError: Cannot coerce cusps from one field to another
```

ABmatrix()

Returns AB-matrix associated to the cusp `self`.

Given R a Dedekind domain and A, B ideals of R in inverse classes, an AB-matrix is a matrix realizing the isomorphism between R+R and A+B. An AB-matrix associated to a cusp [a1: a2] is an AB-matrix with A the ideal associated to the cusp (A=<a1, a2>) and first column given by the coefficients of the cusp.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: alpha = NFCusp(k, oo)
sage: alpha.ABmatrix()
[1, 0, 0, 1]


sage: alpha = NFCusp(k, 0)
sage: alpha.ABmatrix()
[0, -1, 1, 0]
```

Note that the AB-matrix associated to a cusp is not unique, and the output of the `ABmatrix` function may change.

```
sage: alpha = NFCusp(k, 3/2, a-1)
sage: M = alpha.ABmatrix()
sage: M # random
[-a^2 - a - 1, -3*a - 7, 8, -2*a^2 - 3*a + 4]
sage: M[0] == alpha.numerator() and M[2]==alpha.denominator()
True
```

An AB-matrix associated to a cusp alpha will send Infinity to alpha:

```
sage: alpha = NFCusp(k, 3, a-1)
sage: M = alpha.ABmatrix()
sage: (k.ideal(M[1], M[3])*alpha.ideal()).is_principal()
True
sage: M[0] == alpha.numerator() and M[2]==alpha.denominator()
True
sage: NFCusp(k, oo).apply(M) == alpha
True
```

apply(*g*)

Return g(`self`), where `g` is a 2x2 matrix, which we view as a linear fractional transformation.

INPUT:

- `g` – a list of integral elements [a, b, c, d] that are the entries of a 2x2 matrix.

OUTPUT:

A number field cusp, obtained by the action of `g` on the cusp `self`.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 23)
sage: beta = NFCusp(k, 0, 1)
sage: beta.apply([0, -1, 1, 0])
Cusp Infinity of Number Field in a with defining polynomial x^2 + 23
sage: beta.apply([1, a, 0, 1])
Cusp [a: 1] of Number Field in a with defining polynomial x^2 + 23
```

denominator()

Return the denominator of the cusp `self`.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 1)
sage: c = NFCusp(k, a, 2)
sage: c.denominator()
2
sage: d = NFCusp(k, 1, a + 1);d
Cusp [1: a + 1] of Number Field in a with defining polynomial x^2 + 1
sage: d.denominator()
a + 1
sage: NFCusp(k, oo).denominator()
0
```

ideal()

Returns the ideal associated to the cusp `self`.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 23)
sage: alpha = NFCusp(k, 3, a-1)
sage: alpha.ideal()
Fractional ideal (3, 1/2*a - 1/2)
sage: NFCusp(k, oo).ideal()
Fractional ideal (1)
```

is_Gamma0_equivalent(*other, N, Transformation=False*)

Checks if cusps `self` and `other` are $\Gamma_0(N)$- equivalent.

INPUT:

- `other` – a number field cusp or a list of two number field elements which define a cusp.
- `N` – an ideal of the number field (level)

OUTPUT:

- bool – `True` if the cusps are equivalent.
- a transformation matrix – (if `Transformation=True`) a list of integral elements [a, b, c, d] which are the entries of a 2x2 matrix M in $\Gamma_0(N)$ such that M * `self` = `other` if `other` and `self` are $\Gamma_0(N)$- equivalent. If `self` and `other` are not equivalent it returns zero.

EXAMPLES:

```
sage: K.<a> = NumberField(x^3-10)
sage: N = K.ideal(a-1)
sage: alpha = NFCusp(K, 0)
sage: beta = NFCusp(K, oo)
sage: alpha.is_Gamma0_equivalent(beta, N)
False
```

```
sage: alpha.is_Gamma0_equivalent(beta, K.ideal(1))
True
sage: b, M = alpha.is_Gamma0_equivalent(beta, K.ideal(1),Transformation=True)
sage: alpha.apply(M)
Cusp Infinity of Number Field in a with defining polynomial x^3 - 10


sage: k.<a> = NumberField(x^2+23)
sage: N = k.ideal(3)
sage: alpha1 = NFCusp(k, a+1, 4)
sage: alpha2 = NFCusp(k, a-8, 29)
sage: alpha1.is_Gamma0_equivalent(alpha2, N)
True
sage: b, M = alpha1.is_Gamma0_equivalent(alpha2, N, Transformation=True)
sage: alpha1.apply(M) == alpha2
True
```

is_infinity()

Returns `True` if this is the cusp infinity.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 1)
sage: NFCusp(k, a, 2).is_infinity()
False
sage: NFCusp(k, 2, 0).is_infinity()
True
sage: NFCusp(k, oo).is_infinity()
True
```

number_field()

Returns the number field of definition of the cusp `self`

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 2)
sage: alpha = NFCusp(k, 1, a + 1)
sage: alpha.number_field()
Number Field in a with defining polynomial x^2 + 2
```

numerator()

Return the numerator of the cusp `self`.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 1)
sage: c = NFCusp(k, a, 2)
sage: c.numerator()
a
sage: d = NFCusp(k, 1, a)
sage: d.numerator()
1
sage: NFCusp(k, oo).numerator()
1
```

NFCusps(*number_field, use_cache=True*)

The set of cusps of a number field K, i.e. $\mathbb{P}^1(K)$.

INPUT:

- number_field – a number field
- use_cache – bool (default=True) - to set a cache of number fields and their associated sets of cusps

OUTPUT:

The set of cusps over the given number field.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 5)
sage: kCusps = NFCusps(k); kCusps
Set of all cusps of Number Field in a with defining polynomial x^2 + 5
sage: kCusps is NFCusps(k)
True
```

Saving and loading works:

```
sage: loads(kCusps.dumps()) == kCusps
True
```

We test use_cache:

```
sage: NFCusps_clear_cache()
sage: k.<a> = NumberField(x^2 + 11)
sage: kCusps = NFCusps(k, use_cache=False)
sage: sage.modular.cusps_nf._nfcusps_cache
{}
sage: kCusps = NFCusps(k, use_cache=True)
sage: sage.modular.cusps_nf._nfcusps_cache
{Number Field in a with defining polynomial x^2 + 11: ...}
sage: kCusps is NFCusps(k, use_cache=False)
False
sage: kCusps is NFCusps(k, use_cache=True)
True
```

class NFCuspsSpace(*number_field*)

Bases: sage.structure.parent_base.ParentWithBase

The set of cusps of a number field. See NFCusps for full documentation.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 5)
sage: kCusps = NFCusps(k); kCusps
Set of all cusps of Number Field in a with defining polynomial x^2 + 5
```

number_field()

Return the number field that this set of cusps is attached to.

EXAMPLES:

```
sage: k.<a> = NumberField(x^2 + 1)
sage: kCusps = NFCusps(k)
sage: kCusps.number_field()
Number Field in a with defining polynomial x^2 + 1
```

NFCusps_clear_cache()

Clear the global cache of sets of cusps over number fields.

EXAMPLES:

```
sage: sage.modular.cusps_nf.NFCusps_clear_cache()
sage: k.<a> = NumberField(x^3 + 51)
sage: kCusps = NFCusps(k); kCusps
Set of all cusps of Number Field in a with defining polynomial x^3 + 51
sage: sage.modular.cusps_nf._nfcusps_cache.keys()
[Number Field in a with defining polynomial x^3 + 51]
sage: NFCusps_clear_cache()
sage: sage.modular.cusps_nf._nfcusps_cache.keys()
[]
```

NFCusps_clear_list_reprs_cache()

Clear the global cache of lists of representatives for ideal classes.

EXAMPLES:

```
sage: sage.modular.cusps_nf.NFCusps_clear_list_reprs_cache()
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(a+1)
sage: sage.modular.cusps_nf.list_of_representatives(N)
(Fractional ideal (1), Fractional ideal (17, a - 5))
sage: sage.modular.cusps_nf._list_reprs_cache.keys()
[Fractional ideal (a + 1)]
sage: sage.modular.cusps_nf.NFCusps_clear_list_reprs_cache()
sage: sage.modular.cusps_nf._list_reprs_cache.keys()
[]
```

NFCusps_ideal_reps_for_levelN(*N*, *nlists=1*)

Returns a list of lists (`nlists` different lists) of prime ideals, coprime to `N`, representing every ideal class of the number field.

INPUT:

- `N` – number field ideal.
- `nlists` – optional (default 1). The number of lists of prime ideals we want.

OUTPUT:

A list of lists of ideals representatives of the ideal classes, all coprime to `N`, representing every ideal.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(5, a + 1)
sage: from sage.modular.cusps_nf import NFCusps_ideal_reps_for_levelN
sage: NFCusps_ideal_reps_for_levelN(N)
[(Fractional ideal (1), Fractional ideal (2, a + 1))]
sage: L = NFCusps_ideal_reps_for_levelN(N, 3)
sage: all([len(L[i])==k.class_number() for i in range(len(L))])
True
```

```
sage: k.<a> = NumberField(x^4 - x^3 -21*x^2 + 17*x + 133)
sage: N = k.ideal(6)
sage: from sage.modular.cusps_nf import NFCusps_ideal_reps_for_levelN
sage: NFCusps_ideal_reps_for_levelN(N)
[(Fractional ideal (1),
  Fractional ideal (13, a - 2),
  Fractional ideal (43, a - 1),
```

```
  Fractional ideal (67, a + 17))]
sage: L = NFCusps_ideal_reps_for_levelN(N, 5)
sage: all([len(L[i])==k.class_number() for i in range(len(L))])
True
```

list_of_representatives(*N*)

Returns a list of ideals, coprime to the ideal N, representatives of the ideal classes of the corresponding number field.

Note: This list, used every time we check $\Gamma_0(N)$ - equivalence of cusps, is cached.

INPUT:

- N – an ideal of a number field.

OUTPUT:

A list of ideals coprime to the ideal N, such that they are representatives of all the ideal classes of the number field.

EXAMPLES:

```
sage: sage.modular.cusps_nf.NFCusps_clear_list_reprs_cache()
sage: sage.modular.cusps_nf._list_reprs_cache.keys()
[]
```

```
sage: from sage.modular.cusps_nf import list_of_representatives
sage: k.<a> = NumberField(x^4 + 13*x^3 - 11)
sage: N = k.ideal(713, a + 208)
sage: L = list_of_representatives(N); L
(Fractional ideal (1),
Fractional ideal (37, a + 12),
Fractional ideal (47, a - 9))
```

The output of `list_of_representatives` has been cached:

```
sage: sage.modular.cusps_nf._list_reprs_cache.keys()
[Fractional ideal (713, a + 208)]
sage: sage.modular.cusps_nf._list_reprs_cache[N]
(Fractional ideal (1),
Fractional ideal (37, a + 12),
Fractional ideal (47, a - 9))
```

number_of_Gamma0_NFCusps(*N*)

Returns the total number of orbits of cusps under the action of the congruence subgroup $\Gamma_0(N)$.

INPUT:

- N – a number field ideal.

OUTPUT:

ingeter – the number of orbits of cusps under Gamma0(N)-action.

EXAMPLES:

```
sage: k.<a> = NumberField(x^3 + 11)
sage: N = k.ideal(2, a+1)
sage: from sage.modular.cusps_nf import number_of_Gamma0_NFCusps
sage: number_of_Gamma0_NFCusps(N)
```

```
4
sage: L = Gamma0_NFCusps(N)
sage: len(L) == number_of_Gamma0_NFCusps(N)
True
```

units_mod_ideal(*I*)

Returns integral elements of the number field representing the images of the global units modulo the ideal `I`.

INPUT:

- `I` – number field ideal.

OUTPUT:

A list of integral elements of the number field representing the images of the global units modulo the ideal `I`. Elements of the list might be equivalent to each other mod `I`.

EXAMPLES:

```
sage: from sage.modular.cusps_nf import units_mod_ideal
sage: k.<a> = NumberField(x^2 + 1)
sage: I = k.ideal(a + 1)
sage: units_mod_ideal(I)
[1]
sage: I = k.ideal(3)
sage: units_mod_ideal(I)
[1, -a, -1, a]
```

```
sage: from sage.modular.cusps_nf import units_mod_ideal
sage: k.<a> = NumberField(x^3 + 11)
sage: k.unit_group()
Unit group with structure C2 x Z of Number Field in a with defining polynomial x^3
sage: I = k.ideal(5, a + 1)
sage: units_mod_ideal(I)
[1,
2*a^2 + 4*a - 1,
...]
```

```
sage: from sage.modular.cusps_nf import units_mod_ideal
sage: k.<a> = NumberField(x^4 - x^3 -21*x^2 + 17*x + 133)
sage: k.unit_group()
Unit group with structure C6 x Z of Number Field in a with defining polynomial x^4
sage: I = k.ideal(3)
sage: U = units_mod_ideal(I)
sage: all([U[j].is_unit() and not (U[j] in I) for j in range(len(U))])
True
```

A.2 Implementation of the algorithms in Chapter 3

Below we list the current version of the author's implementation of the algorithms described in §3.2. We give a small example to illustrate how it works.

Example A.2.1. We compute the data in Table 3.2 for the field $K = \mathbb{Q}(\sqrt{-14})$.

```
sage: attach './FundDomains.sage'

sage: k.<a> = NumberField(x^3 + 14)
sage: S, V = fundamental_domain(k)

sage: max_mu = max([s[1] for s in S]); max_mu
56

sage: min_v = min([v[2] for v in V]); min_v
1/108
```

Listing 1: FundDomains.sage

```
def fundamental_domain(k):
    """
    For an imaginary quadratic field K, the function returns
    two lists, S and V. S is the list of hemispheres that form
    the floor of a fundamental domain. V is the list of vertices
    coming from the intersection of 3 or more hemispheres.
    """
    S = initial_list(k)
    V, S = clean_list(k, S)
    V, S = Swan(k, S, V)
    return S, V

#------ Plotting functions ------------------------------

def plot_FunDomain_projection(k, S, V, Full=True):
    if Full == True:
        BS = []
        BV = []
        for s in S:
            BS.append(s)
            if s[0][0]>0:
                new_s = (k((-s[0][0],s[0][1])), s[1])
                BS.append(new_s)
        for v in V + [(s[0], s[1], 0) for s in sk]:
            BV.append(v)
            if v[0]>0:
                new_v = (-v[0], v[1], v[2])
                BV.append(new_v)
```

```
    L = []
    D = k.absolute_generator().norm()
    w = RR(D).sqrt()
    for s in BS:
        centre = (s[0][0], s[0][1]*w)
        rad = 1/RR(s[1]).sqrt()
        L.append(circle(centre, rad, aspect_ratio=1, \
                 fill=True, alpha = 0.2, rgbcolor = 'blue'))
    for v in BV:
        P = (v[0], v[1]*w)
        L.append(point(P, rgbcolor = 'red', pointsize=22))
    proj = sum(L)
    if ZZ(D).mod(4) == 3:
        b = 4
    else:
        b = 2
    print b
    proj.set_axes_range(-0.5, 0.5, 0, w/b)
    return proj

def plot_Bianchi_diagram(k, S):
    BS = []
    for s in S:
        BS.append(s)
        if s[0][0]>0:
            new_s = (k((-s[0][0],s[0][1])), s[1])
            BS.append(new_s)
    X, Y, Z = var('X, Y, Z')
    L = []
    D = k.absolute_generator().norm()
    w = RR(D).sqrt()
    b = 2
    if ZZ(D).mod(4)==3:
        b = 4
    Yrange = ceil(w/b)
    for s in BS:
        c = k(s[0])
        eq = (X - c[0])^2 + (Y - c[1]*w)^2 + Z^2 - 1/s[1]
        L.append(implicit_plot3d(eq, (Y, 0, Yrange ), (X, -0.5, 0.5),
                                 \(Z, 0, 1), plot_points=60, aspect_ratio=1))
    return sum(L)

#-----------------------------------------------------------------------
#----- Finding an initial list of hemispheres -------------------------

#----- Auxiliar functions ---------------------------------

@cached_function
def elements_of_norm(k,i):
    """
    Returns a list of elements of k of norm i, for k=Q(sqrt(-d)) imaginary
    quadratic field with d>3. The list contains both 'x' and '-x'.
    """
    L = []
    for l in k.elements_of_norm(i):
```

```
            L.append(I)
            L.append(-I)
        return L

    def list_of_bounded_elems(k, minB, maxB):
        """
        Returns a list of elements of k with norm in the interval [minB, maxB]
        (as before k=Q(sqrt(-d)) imaginary quadratic field with d>3).
        """
        L = []
        for i in range(minB + 1,maxB + 1):
            L = L + elements_of_norm(k,i)
        return L


    def in_circle(k, P, s, OnlyInside=False):
        """
        Returns True if the point P = (P0, P1) (coordinates in k basis) is
        inside the circle s (where s = [alpha, i] with alpha in k the centre
        of the circle, i in ZZ the square of the inverse of the radius).

        Optional parameter 'OnlyInside' to specify if we want strict inclusion.
        """
        rad = 1/s[1] #this rad is the square of the radius, and is rational
        if OnlyInside:
            try:
                return (k(P)-s[0]).norm() < rad
            except TypeError:
                D = k.absolute_generator().norm()
                return ((s[0][0]-P[0])^2 + (s[0][1]*sqrt(D) - P[1])^2)< rad
        else:
            try:
                return (k(P)-s[0]).norm() <= rad
            except TypeError:
                D = k.absolute_generator().norm()
                return ((s[0][0]-P[0])^2 + (s[0][1]*sqrt(D) - P[1])^2)<= rad

    def find_circles(k, P, S, OnlyInside=False):
        """
        Given 'P' a point and 'S' a list of circles,returns a list of the
        circles in S that contain the point 'P'.
        Optional argument 'OnlyInside' to specify strict inclusion.
        """
        S0 = []
        for s in S:
            if in_circle(k, P, s, OnlyInside):
                S0.append(s)
        return S0

    def hem_covered(k, s, L):
        """
        Given 's' hemisphere, 'L' list of hemispheres, returns 'True' if 's' is
        contained in one of the hemispheres of 'L'.
        """
        rad = 1/sqrt(s[1])
```

```
    for l in L:
        #we first check if the centre of 's' is contained in the
        #hemisphere 'l' of L (in fact in the projection of 'l'
        #on the plane).
        if in_circle(k, s[0], l, OnlyInside=True):
            rad_l = 1/sqrt(l[1])
            if (s[0] - l[0]).norm() <= (rad - rad_l)^2:
                return True
    return False


#------ Main functions to list hemispheres ----------------------

def initial_list(k):
    """
    Returns a list of hemispheres that cover the fundamental rectangle
    on the floor of the hyperbolic 3-space.
    """
    dk = k.discriminant()
    if dk.mod(4) == 1:
        D = dk.abs()
        b = 4 #fund region is [0, 1/2] x[0, sqrt(D)/b]
    else:
        D = (dk/4).abs()
        b = 2
    #we initialize the list of hemispheres with the only principal
    #hemispheres with radius 1
    if b==2:
        L = [(k(0), 1), (k(1), 1), (k((0,1)),1)]
    else:
        L = [(k(0), 1), (k(1), 1), (k((1/2,1/2)),1)]
    #num = [0] + list_of_bounded_elems(k, 0, 1 + D)
    num = list_of_bounded_elems(k, 0, 1 + D)
    #L1, num = list_for_radius_i(k, 1, num, L)
    #L = L + L1
    #i = 2
    if k.class_number()==1: #for h=1 I use a different recursion
        L = check_rectangle(k, (0, 0), 1/2, 1/b, num, L)
    else:
        i = 2
        Sing = singular_points_in_F(k)
        check = False
        Pcov = []
        while check==False: #we loop over the size of the radius
            Li, num = list_for_radius_i(k, i, num, L)
            if Li:
                L = L + Li
                if not check:
                    check = True
                    for P in Sing:
                        if not (P in Pcov):
                            if find_circles(k, P, Li):
                                checkP = check_cover_for_sing_P(k, P, L)
                            else: # not adding new circles through P
                                checkP = False
```

```
                            if checkP:
                                Pcov.append(P)
                            check = check and checkP
                    i = i + 1
                R = list_rectangles(k, Sing, L)
                L = check_rectangles(k, i, num, R, L) #breadth-first recursion check
        return L

    def list_for_radius_i(k, i, num, S, R = None):
        """
        Given a list of hemispheres 'S' and an integer 'i', return a list of
        hemispheres of radius sqrt(1/i) which are not covered by any of the
        bigger hemispheres already in 'S' and that intersect the rectangle 'R'.

        By default 'R' is the fundamental rectangle.
        I update the list of possible numerators (for the centres,
        given by cusps)
        """
        L = []
        dk = k.discriminant()
        if dk.mod(4) == 1:
            D = dk.abs()
            b = 4 #fund region is [0, 1/2] x[0, sqrt(D)/b]
        else:
            D = (dk/4).abs()
            b = 2
        if not R:
            R = ((0, 0), 1/2, 1/b)
        w = RR(D).sqrt()
        rad = RR(1/i).sqrt() #same radius since denominators have same norm=i
        den = elements_of_norm(k,i)
        den = [l for l in den if l[1]>0 or (l[1]==0 and l[0]>=0)]
        n = k(num[-1]).norm()
        if n < i*(1 + D):
            num = num + list_of_bounded_elems(k, n, i*(1 + D))
        for y in den: #we add all hemispheres of radius rad...
            for x in num:
                alpha = x/y
                if (-1 <= alpha[0] <= 1) and (0 <= alpha[1] <=1):
                    if k.ideal(x, y).norm()==1:
                        if (-rad + R[0][0]) <= alpha[0]
                                                \<= (R[0][0] + R[1] + rad):
                            if (-rad + R[0][1]*w) <= alpha[1]*w <= \
                               (rad + (R[0][1] + R[2])*w):
                                if not hem_covered(k, (alpha, i), S):
                                    L.append((alpha, i))
                                    S = S + L
        return L, num

    #----- Functions concerning singular points ----------------------

    # Set of singular points ------------------------------------------

    def singular_points(k):
        """
```

```
    Returns a list of representatives for the singular points modulo
    translations by elements of the ring of integers of the field.

    Algorithm: Swan.
    """
    L = []
    dk = k.discriminant()
    if dk.mod(4) == 1:
        D = dk.abs()
    else:
        D = (dk/4).abs()
    if ZZ(D).mod(4) == 3:
        s = 4
        step = 2
    else:
        s = 2
        step = 1
    while (3/4)*s^2<=D:
        for r in range(-s/2 + 1, s/2 + 1):
            if s^2<=(r^2 + D):
                if (step*s).divides(r^2 + D):
                    s0 = ZZ(s/step)
                    for p in range(s0):
                        if ZZ(p).gcd(s0)==1:
                            L.append((p*r/s, p/s))
        s = s + step
    return L

def singular_points_in_F(k):
    """
    Returns a list with the singular points which are on the floor of the
    fundamental region.
    """
    L = singular_points(k)
    LinF = []
    dk = k.discriminant()
    if dk.mod(4)==1:
        h = 1/4
        d = 2
    else:
        h = 1/2
        d = 1
    for P in L: #we test if P or a translated P is inside Fk
        sign = 1
        t_rang = xrange(P[0] - 1/2, P[0] + 1)
        if P[0]<0:
            sign = -1
            t_rang = xrange(sign*(P[0]), sign*(P[0] - 1/2) + 1)
        for k in range(-d*(h - P[1]), d*P[1] + 1):
            if 0< (P[1] - k/d) <= h:
                for t in t_rang:
                    if k/d <= (P[0] - sign*t) <= (1/2 + k/d):
                        LinF.append((P[0] - sign*t - k/d, P[1] - k/d))
    return LinF
```

```
#----- Checking covering of singular points (all is 2-dimensional)-----

def check_cover_for_sing_P(k, P, S):
    """
    Returns True if the singular point P is totally surrounded by
    circles (through P).
    """
    D = k.absolute_generator().norm()
    S0 = find_circles(k, P, S)
    P = k(P)
    # first find the list of angles around which the point P is covered
    L = []
    for s in S0:
        if P[1]==s[0][1]:
            m = oo
        else:
            m = -(P[0]-s[0][0])/((P[1]-s[0][1])*sqrt(D))
        zeta = arctan(m)
        if m==0:
            if P[1]>s[0][1]:
                L.append((pi, 2*pi))
            else:
                L.append((0, pi))
        else:
            if m>0:
                if P[0]<s[0][0]:
                    L = L + [(pi + zeta, 2*pi + 1), (-1, zeta)]
                else:
                    L.append((zeta, zeta + pi))
            else:
                if P[0]<s[0][0]:
                    L = L + [(2*pi + zeta, 2*pi +1), (-1, pi + zeta)]
                else:
                    L.append((pi + zeta, 2*pi + zeta))

    # now we check if our list of angles covers all of [0, 2*pi]
    bmax = 0
    while bmax<=2*pi:
        cov = 0
        for j in L:
            if j[0]<bmax and j[1]>bmax:
                bmax = j[1]
                cov = 1
        if cov==0:
            return False
    return True

#----- Intersection of 'principal' circles surrounding a singular point

def intersection_point(k, P, c1, c2):
    """
    P singular point, c1, c2 circles through P.
    Returns other point of intersection of c1 and c2 (different from P)
    """
    D = k.absolute_generator().norm()
```

```
    rad1 = 1/c1[1] #square of the radius in fact
    rad2 = 1/c2[1]
    P = k(P)
    a = c2[0][0] - c1[0][0]
    b = c2[0][1] - c1[0][1]
    c = 1/2*(rad1 - rad2 + c2[0][0]^2 - c1[0][0]^2 + \
        (c2[0][1]^2 - c1[0][1]^2)*D)
    if a.is_zero():
        y = c/(b*D)
        B = -2*c1[0][0]
        x = -B - P[0]
        return (x, y)
    else:
        A = D*(b/a)^2 + 1
        B = 2*(c1[0][0]*b/a - b*c/a^2 - c1[0][1])
        y = -B/A - P[1]
        return (1/a*(c-b*y*D) , y)

def intersection_singular_circles(k, P, c1, c2):
    """
    P singular point, c1, c2 circles through P.
    Checks if c1 and c2 intersect and not only touch at P, and then
    returns the intersection point.
    """
    rad1 = 1/c1[1]
    rad2 = 1/c2[1]
    # we are only interested in circles that intersect in more than
    # one point (=sing point)
    if k(c1[0] - c2[0]).norm() < rad1 + rad2 + 2*sqrt(rad1*rad2):
        return intersection_point(k, P, c1, c2)
    else:
        return ()

def intersection_sing_point(k, P, S):
    """
    P singular point, S list of circles.
    Returns all points of intersection (except trivial P) between
    the circles of S that go through P.
    """
    s = find_circles(k, P, S)
    L = []
    Ldone = []
    for c1 in s:
        Ldone.append(c1)
        for c2 in s:
            if not (c2 in Ldone):
                I = intersection_singular_circles(k, P, c1, c2)
                if I and not I==P: #catch wrong intersec point
                    L.append(I)
    return L

#------ Rectangles around singular points ----------------------

def rect_around_sing_point(k, sk, S):
    """
```

```
For a singular point sk, returns a rectangle around sk which we can
guarantee is covered by the circles in the set S.
The rectangle is given in the form (Point, width, height).
"""
dk = k.discriminant()
if dk.mod(4) == 1:
    h = 1/4 #fund region is [0, 1/2] x[0, sqrt(D)*h]
else:
    h = 1/2
L = intersection_sing_point(k, sk, S)
# now find intersection point inside fundamental region at minimum
# distance to sk
minP = 0
P = ()
for u in L:
    if 0<=u[0]<=1/2 and 0<=u[1]<=h:
        if minP==0:
            P = u
            minP = (k(sk) - k(P)).norm()
        else:
            d = (k(sk) - k(u)).norm()
            if d < minP:
                minP = d
                P = u
# if all intersection points fall out of the fundamental region:
if not P:
    maxP1 = max([v[1] for v in L])
    minP1 = min([v[1] for v in L])
    if maxP1 > h:
        return ((0, minP1), 1/2, h - minP1)
    else:
        if minP1 > 0:
            return ((0, minP1), 1/2, maxP1 - minP1)
        else:
            return ((0, 0), 1/2, maxP1)

# in most cases at least one of the intersection points is inside the
# fundamental region; then we find the the rectangle as follows:
wP = 2*(sk[0] - P[0])
hP = 2*(sk[1] - P[1])
if wP > 0:
    r1 = P[0]
    # adjust the rectangle in case crosses out of fundamental region
    if (r1 + wP) > 1/2:
        wP = 1/2 - r1
else:
    wP = -wP
    r1 = P[0] - wP
    if r1 < 0: # adjusting rectangle
        wP = r1 + wP
        r1 = 0
if hP > 0:
    r2 = P[1]
    if (r2 + hP) > h:
        hP = h - r2
```

```
    else:
        hP = -hP
        r2 = P[1] - hP
        if r2 < 0:
            hP = r2 + hP
            r2 = 0
    return (r1, r2), wP, hP

#------ Checking rectangles ------------------------------------------
#------ Checking if a list of circles (hemispheres) covers all of F_K

def rectangle_cov(k, r, S):
    """
    Checks if rectangle r is covered by the circles in S.
    Returns 0 if any corner of r is not covered, 1 if all corners are
    covered (by different circles) and 2 if all corners covered by one
    circle from S (so the whole of r is covered).
    """
    c = find_circles(k, r[0], S)
    if not c:
        return 0 # = 0: need more hemispheres, v is not covered at all!
    vv = [(r[0][0] + r[1], r[0][1]), (r[0][0], r[0][1] + r[2]), \
          (r[0][0] + r[1], r[0][1] + r[2])]
    check = 2
    for u in vv:
        if check == 1: #two vertices at least not covered by same hemisph
            if not find_circles(k, u, S):
                return 0 #that is, this vertex not covered at all!
        else:
            c_u = [s for s in c if in_circle(k, u, s)]
            if not c_u:
                #vertex u not in any circle s containing v =>c_u empty
                check = 1
                if not find_circles(k, u, S): #no circle contains v
                    return 0
            else:
                c = c_u
    return check #if check=2, whole rectangle covered by one circle

def check_rectangle(k, v, w, h, num, S):
    """
    Checks if rectangle of width 'w', height 'h' is covered by the
    circles of the set S.
    Adds more hemispheres to S until the rectangle is covered.
    We only use this function for fields with class number 1.
    """
    check = rectangle_cov(k, (v, w, h), S)
    if check==2:
        return S
    if check==0:
        i = S[-1][1] #last radius in S
        while check==0:
            Li, num = list_for_radius_i(k, i + 1, num, S)
            if Li: #we have added something new
                S = S + Li
```

```
                check = rectangle_cov(k, (v, w, h), S)
            i = i + 1
        if check == 2:
            return S
    if check==1:
        R0 = rectangle_subdivision(k, (v, w, h))
        for r in R0:
            Sr = check_rectangle(k, r[0], r[1], r[2], num, S)
            S = Sr
        return S

def check_rectangles(k, i, num, R, S):
    """
    Checks if all rectangles in the list R are covered by the circles
    of the set S. Adds more hemispheres to S until all rectangles are
    covered. We use this function for fields with class number > 1,
    where previous (depth) recursive algorithm doesn't work in general.
    """
    if not R:
        return S
    R0 = []
    for r in R:
        check = rectangle_cov(k, r, S)
        if check == 1:
            R0 = R0 + rectangle_subdivision(k, r)
        if check == 0:
            norm_d = i
            while check==0:
                #print "checking r = ", r
                Li, num = list_for_radius_i(k, norm_d + 1, num, S, r)
                if Li: #we have added something new
                    #print "adding ", Li
                    S = S + Li
                    check = rectangle_cov(k, r, S)
                norm_d = norm_d + 1
            if check == 1:
                R0 = R0 + rectangle_subdivision(k, r)
    return check_rectangles(k, i, num, R0, S)

def rectangle_subdivision(k, r):
    """
    Returns list of rectangles (roughly squares) that gives a subdivision
    of rectangle 'r'
    """
    R = []
    sqD = RR(k.absolute_generator().norm()).sqrt()
    P, wP, hP = r
    if hP*sqD > wP:
        t = 2
        th = t*floor(hP*sqD/wP)
    else:
        th = 2
        t = 2*floor(wP/(hP*sqD))
    rw = wP/t
    rh = hP/th
```

```
    for j in range(th):
        for i in range(t):
            u1 = P[0] + i*rw
            u2 = P[1] + j*rh
            R.append(((u1, u2), rw, rh))
    return R

def list_rectangles(k, sk, S):
    """
    Returns a list of rectangles that contain all the areas of fundamental
    region that need checking (excludes surroundings of singular points).
    """
    dk = k.discriminant()
    if dk.mod(4) == 1:
        D = dk.abs()
        h = 1/4 #fund region is [0, 1/2] x[0, sqrt(D)*h]
    else:
        D = (dk/4).abs()
        h = 1/2
    R = []
    R = R + subdivide_rect(k, (0, 0), 1/2, h, sk, S)
    return R

def subdivide_rect(k, P, wP, hP, sk, S):
    """
    P, wP, hP a rectangle; sk set of singular points; S set of circles.
    Subdivides rectangle (P, wP, hP) into rectangles that don't contain
    singular points.
    """
    R = []
    L = [(rect_around_sing_point(k, s, S)) for s in sk]
    num, sing = number_sing(k, P, wP, hP, L)
    if num == 0:
        R.append((P, wP, hP))
    if num == 1:
        R = R + one_subdiv(k, P, wP, hP, sing[0], S)
    if num > 1:
        R0 = rectangle_subdivision(k, (P, wP, hP))
        for r in R0:
            R = R + subdivide_rect(k, r[0], r[1], r[2], sk, S)
    return R

def number_sing(k, p, wp, hp, Lr):
    """
    p, wp, hp a rectangle; Lr list of neighborhoods of singular points.
    Returns number of singular points and list of rectangles around them
    that cut the given rectangle (p, wp, hp)
    """
    n = 0
    L = [] # list of the neighborhoods inside our main rectangle
    for r in Lr:
        if (p[0]<=r[0][0]<(p[0]+wp)) or (p[0]<(r[0][0] + r[1])<(p[0]+wp)):
            if (p[1]<r[0][1]<(p[1]+hp)) or \
                                    (p[1]<(r[0][1] + r[2])<(p[1]+hp)):
                n = n + 1
```

```
                L.append(r)
    return n, L

def one_subdiv(k, P, wP, hP, rk, S):
    """
    Return subdivision of a rectangle (P, wP, hP) which includes only one
    singular rectangule rk.
    """
    R = []
    u, wu, hu = rk[0], rk[1], rk[2]
    w0 = 0
    if u[0] > P[0]:
        w0 = u[0] - P[0]
        R.append(((P[0], P[1]), w0, hP))
    if (u[0] + wu) < (P[0] + wP):
        w1 = (u[0] + wu) - (P[0] + w0)
        R.append(((P[0] + w0 + w1, P[1]), (P[0] + wP) - (u[0] + wu), hP))
    else:
        w1 = (P[0] + wP) - (P[0] + w0)
    if u[1] > P[1]:
        R.append(((P[0] + w0, P[1]), w1, u[1] - P[1]))
    if (u[1] + hu) < (P[1] + hP):
        R.append(((P[0] + w0, u[1] + hu), w1, (P[1] + hP) - (u[1] + hu)))
    return R

#-------------------------------------------------------------
#----- Cleaning list of redundant hemispheres ----------------

#----- Intersections of 3 hemispheres ------------------------

def circles_intersection_check(k, c1, c2):
    """
    Checks if circles c1, c2 intersect; returns True or False.
    We assume circles are not included one inside the other.
    """
    rad1 = 1/c1[1]
    rad2 = 1/c2[1]
    c_distance = sqrt(k(c1[0]-c2[0]).norm())
    if RR(c_distance) < RR(sqrt(rad1)) + RR(sqrt(rad2)):
        return True
    else:
        return False

def hemispheres_intersection(k, s1, s2, s3):
    """
    Checks if hemispheres s1, s2, s3 truly intersect (assumes that they
    intersect pairwise), and if any of them happens to be covered by the
    union of the other two. Returns redundant hemisphere (if any) and
    point of intersection (if any).
    """
    a1 = s2[0][0] - s1[0][0]
    b1 = s2[0][1] - s1[0][1]
    a2 = s3[0][0] - s1[0][0]
    b2 = s3[0][1] - s1[0][1]
    if (b1.is_zero() and b2.is_zero()) or (a1.is_zero() and a2.is_zero()) \
```

```
            or (b2*a1==b1*a2):
            s = check_redundant_hem(k, s1, s2, s3)
            return s, ()
        D = k.absolute_generator().norm()
        rad1 = 1/s1[1]
        rad2 = 1/s2[1]
        rad3 = 1/s3[1]
        c1 = 1/2*(rad1 - rad2 + s2[0][0]^2 - s1[0][0]^2 \
                              + (s2[0][1]^2 - s1[0][1]^2)*D)
        c2 = 1/2*(rad1 - rad3 + s3[0][0]^2 - s1[0][0]^2 \
                              + (s3[0][1]^2 - s1[0][1]^2)*D)
        if a1.is_zero(): #then a2 nonzero
            yP = c1/(b1*D)
            xP = 1/a2*(c2 - b2*D*yP)
        else: #a1 not zero
            yP = (c2 - c1*a2/a1)/(D*(b2 - a2*b1/a1))
            xP = 1/a1*(c1 - b1*D*yP)
        if not all([in_circle(k, (xP, yP), s) for s in [s1, s2, s3]]):
            return (), ()
        tP = rad1 - (k((xP, yP)) - s1[0]).norm()
        return (), (xP, yP, tP)

    def check_redundant_hem(k, s1, s2, s3):
        """
        Given hemispheres s1, s2, s3 with centres in same line, the function
        checks if any of the 3 hemispheres is covered by the other two
        (by checking the position of intersection lines).
        """
        D = k.absolute_generator().norm()
        L = [((s[0][0], s[0][1]),s[1]) for s in [s1, s2, s3]]
        L.sort()
        s1, s2, s3 = L[0], L[1], L[2]
        b12 = s2[0][1] - s1[0][1]
        b23 = s3[0][1] - s2[0][1]
        rad1 = 1/s1[1]
        rad2 = 1/s2[1]
        rad3 = 1/s3[1]
        c12 = 1/2*(rad1 - rad2 + s2[0][0]^2 - s1[0][0]^2 \
                               + (s2[0][1]^2 - s1[0][1]^2)*D)
        c23 = 1/2*(rad2 - rad3 + s3[0][0]^2 - s2[0][0]^2 \
                               + (s3[0][1]^2 - s2[0][1]^2)*D)

        if b12.is_zero(): #horizontal line
            a12 = s2[0][0] - s1[0][0]
            a23 = s3[0][0] - s1[0][0]
            if c23/a23 <= c12/a12:
                return (k((s2[0][0], s2[0][1])), s2[1])
            else:
                return ()
        a12 = s2[0][0] - s1[0][0]
        if (-a12/b12) > 0:
            if c23/b23 >= c12/b12:
                return (k((s2[0][0], s2[0][1])), s2[1])
            else:
                return ()
```

```
    else:
        if c23/b23 <= c12/b12:
            return (k((s2[0][0], s2[0][1])), s2[1])
        else:
            return()
    return ()

#------ the 'cleaning' function...

def clean_list(k, S):
    """
    Cleans list of hemispheres S of unnecessary ones.
    Returns the list of points of 3-intersection and the list
    of hemispheres.
    """
    dk = k.discriminant()
    if dk.mod(4) == 1:
        h = 1/4 #fund region is [0, 1/2] x[0, sqrt(D)*h]
    else:
        h = 1/2
    Sing = singular_points_in_F(k)
    L = S[:]
    IntPoints = []
    for s1 in L:
        check = True
        checkSing = False
        if s1 in S:
            nP = 0
            Ldone = [s1]
            for s2 in L:
                if s2 in S:
                    Ldone.append(s2)
                    if (not(s2==s1)) and \
                                circles_intersection_check(k, s1, s2):
                        Laux = [s for s in L if s in S and not s in Ldone]
                        for s3 in Laux:
                            if circles_intersection_check(k, s1, s3) and \
                                    circles_intersection_check(k, s2, s3):
                                red, P = \
                                   hemispheres_intersection(k, s1, s2, s3)
                                if red and red in S:
                                    S.remove(red)
                                elif P: #no redundant hemispheres
                                    if P in IntPoints:
                                        check = False
                                    elif P[2]==0:
                                        if (P[0], P[1]) in Sing:
                                            checkSing = True
                                    else:
                                        sP = find_hemispheres(k, P, S)
                                        if not sP: #nothing covers P
                                            check = False #we need s1
                                            IntPoints.append(P)
                                        else:
                                            nP = nP + 1
```

```
            if checkSing and nP==0:
                check = False
            if check and (s1 in S):
                S.remove(s1)
    return IntPoints, S

def find_hemispheres(k, P, S, OnSurface=False):
    """
    Finds hemispheres in S that cover the point P.
    """
    L = []
    sP = find_circles(k, (P[0], P[1]), S)
    for s in sP:
        t = 1/s[1] - (k((P[0], P[1])) - s[0]).norm()
        if OnSurface:
            if t==P[2]:
                L.append(s)
        else:
            if t>P[2]:
                L.append(s)
    return L

#-----------------------------------------------------------
#----- Swan's algorithm ------------------------------------

def find_principal_hemisphere(k, P):
    """
    Finds a principal hemisphere that covers P with maximum
    height over P.
    """
    den = list_of_bounded_elems(k, 0, 1/P[2])
    den = [l for l in den if l[1]>0 or (l[1]==0 and l[0]>=0)]
    z = k((P[0], P[1]))
    B = (1 + RR((z*den[-1]).norm()).sqrt())^2
    lam = [0] + list_of_bounded_elems(k, 0, B.ceil())
    tmax = 0
    sP = ()
    for mu in den:
        normmu = k(mu).norm()
        if 1/normmu < tmax:
            return sP
        Bmu = (((RR(1 - normmu*P[2])).sqrt() \
                            + RR((z*mu).norm()).sqrt())^2).ceil()
        for l in [l for l in lam if k(l).norm()<Bmu]:
            if k.ideal(l, mu).norm()==1:
                if (mu*z - l).norm() + normmu*P[2] < 1:
                    t = 1/normmu - (z - l/mu).norm()
                    if t > tmax:
                        tmax = t
                        sP = (k(l/mu), normmu)
    return sP

def Swan(k, S, V=None):
    """
    Swan's algorithm.
```

```
    Returns the list of hemispheres that give the floor of the
    fundamental region, and a list of all the intersection points
    (they are potential vertices).
    """
    LVchecked = []
    LVremove = []
    while True:
        check = True
        V = clear_inter_list(k, V)
        for v in V:
            if check and not v in LVchecked:
                if 0<=len(find_hemispheres(k, v, S, OnSurface=True))<3:
                    #the point shouldnt really be here
                    LVchecked.append(v)
                    LVremove.append(v)
                else:
                    #print "find hemisph for v", v
                    sv = find_principal_hemisphere(k, v)
                    if sv:
                        v0 = v
                        if not sv in S:
                            S.append(sv)
                        check = False
                    else:
                        LVchecked.append(v)
        if check:
            break
        #print "checking point v0", v0
        V, S = adjust_list(k, v0, V, S)
    V = [v for v in V if not v in LVremove]
    return V, S

def clear_inter_list(k, V):
    """
    Clears the list of intersection points of symmetries (to avoid
    checking things twice).
    """
    dk = k.discriminant()
    if dk.mod(4) == 1:
        h = 1/4
    else:
        h = 1/2
    L = V[:]
    for v in V:
        if v[1]<=h:
            if v[0]<0:
                if [u for u in V if u[0]==-v[0] and u[1]==v[1]]:
                    L.remove(v)
            if v[0]>1/2:
                if [u for u in V if u[0]==1 - v[0] and u[1]==v[1]]:
                    L.remove(v)
        else:
            if 0<=v[0]<=1/2:
                if h == 1/2: # we have symmetry
                    if [u for u in V if u[0]==v[0] and u[1]==2*h - v[1]]:
```

```
                        L.remove(v)
                else: # we have glide reflection
                    if [u for u in V if u[0]==(1/2-v[0]) and \
                                        u[1]==2*h - v[1]]:
                        L.remove(v)
            else:
                L.remove(v)
    return L

def adjust_list(k, v0, V, S):
    """
    After adding a hemisphere that covers the point v0, we check again
    our list S for new redundancies resulting from that addition.
    """
    dk = k.discriminant()
    if dk.mod(4) == 1:
        h = 1/4 #fund region is [0, 1/2] x[0, sqrt(D)*h]
    else:
        h = 1/2
    Sv = [s for s in S if in_circle(k, k((v0[0], v0[1])), s)]
    LRed = []
    Ldone = []
    NewPoints = []
    for s1 in Sv:
        nP = 0 #number of intersection points for s1 that are covered
        checkSing = False
        check = True
        if s1 in S:
            Ldone = [s1]
            for s2 in S:
                Ldone.append(s2)
                if (not (s2==s1)) and \
                            circles_intersection_check(k, s1, s2):
                    Laux = [s for s in S if s in S and not s in Ldone]
                    for s3 in Laux:
                        if circles_intersection_check(k, s1, s3) and \
                                circles_intersection_check(k, s2, s3):
                            red, P = \
                                hemispheres_intersection(k, s1, s2, s3)
                            if red:
                                LRed.append(red)
                            elif P:
                                #no redundant hemispheres and intersect
                                if P in NewPoints:
                                    check = False
                                elif P[2]==0:
                                    if 0<=P[0]<=1/2 and 0<=P[1]<=h:
                                        checkSing = True
                                else:
                                    sP = find_hemispheres(k, P, S)
                                    if not sP: #no spheres covering P
                                        check = False #we need to keep s1
                                        if not P in V:
                                            NewPoints.append(P)
                                    else:
```

```
                                        nP = nP + 1
                                        #nP = number of covered points
                                        if P in V:
                                        #P not true vertex
                                            V.remove(P)
        if checkSing and nP==0:
        #all other intersection points except singular one are uncovered
        #this check is necessary for spheres that only intersect at
        #singular points
            check = False
        if check:
            #S.remove(s1)
            LRed.append(s1)
    for s in LRed:
        if s in S:
            S.remove(s)
    V = V + NewPoints
    return V, S
```

Bibliography

[1] M. Akbas and D. Singerman. The normalizer of $\Gamma_0(N)$ in $PSL(2,\mathbb{R})$. *Glasgow Mathematical Journal*, 32(03):317–327, 1990.

[2] A. Atkin and J. Lehner. Hecke operators on $\Gamma_0(m)$. *Mathematische Annalen*, 185(2):134–160, 1970.

[3] F. Bars. The group structure of the normalizer of $\Gamma_0(N)$ after Atkin–Lehner. *Communications in Algebra*, 36(6):2160–2170, 2008.

[4] L. Bianchi. Sui gruppi di sostituzioni lineari con coefficienti appartenenti a corpi quadratici immaginari. *Mathematische Annalen*, 40(3):332–412, 1892. ISSN 0025-5831.

[5] W. Bosma, J. Cannon, and C. Playoust. The Magma algebra system. I. The user language. *J. Symbolic Comput.*, 24(3-4):235–265, 1997.

[6] J. S. Bygott. *Modular Forms and Modular Symbols over Imaginary Quadratic Fields*. PhD thesis, University of Exeter, 1998.

[7] H. Cohen. *A Course in Computational Algebraic Number Theory*. Springer-Verlag, 1999.

[8] J. Conway and S. Norton. Monstrous moonshine. *Bulletin of the London Mathematical Society*, 11(3):308–339, 1979.

[9] J. E. Cremona. *Modular Symbols*. PhD thesis, Oxford University, 1981.

[10] J. E. Cremona. Hyperbolic tessellations, modular symbols, and elliptic curves over complex quadratic fields. *Compositio Math*, 51(3):275–323, 1984.

[11] J. E. Cremona. On $GL(n)$ of a Dedekind domain. *The Quarterly Journal of Mathematics*, 39(4):423–426, 1988.

[12] J. E. Cremona. Modular symbols for $\Gamma_1(N)$ and elliptic curves with everywhere good reduction. *Mathematical Proceedings of the Cambridge Philosophical Society*, 111(2):199–218, 1992.

[13] J. E. Cremona. *Algorithms for modular elliptic curves*. Online edition, 1993.

[14] L. Dembélé. Explicit Computations of Hilbert Modular Forms on $\mathbb{Q}(\sqrt{5})$. *Experimental Mathematics*, 14(4):457–466, 2005.

[15] L. Dembélé. Quaternionic Manin symbols, Brandt matrices, and Hilbert modular forms. *Mathematics of Computation*, 76(258):1039, 2007. ISSN 0025-5718.

[16] J. Elstrodt, F. Grunewald, and J. Mennicke. *Groups acting on hyperbolic space: Harmonic analysis and number theory*. Springer Verlag, 1998.

[17] A. Fröhlich and M. Taylor. *Algebraic number theory*. Cambridge University Press, 1993.

[18] P. Gunnells and D. Yasaki. Hecke operators and Hilbert modular forms. *Proceedings of the 8th international conference on Algorithmic number theory*, pages 387–401, 2008.

[19] P. E. Gunnells. Modular Symbols for Q-Rank One Groups and Voronoĭ Reduction. *Journal of Number Theory*, 75(2):198–219, 1999.

[20] G. Humbert. Sur la réduction des formes d'Hermite dans un corps quadratique imaginaire. *Compt. Rend. Acad. Sci. Paris*, 16:189–196, 1915.

[21] J. Lehner and M. Newman. Weierstrass Points of $\Gamma_0(n)$. *Annals of Mathematics*, 79(2):360–368, 1964.

[22] M. P. Lingham. *Modular forms and elliptic curves over imaginary quadratic fields*. PhD thesis, University of Nottingham, 2005.

[23] M. Newman. The normalizer of certain modular subgroups. *Canadian Journal of Mathematics*, 8:29–31, 1956.

[24] A. Rahm. *(Co)homologies and K-theory of Bianchi groups using computational geometric models*. PhD thesis, Université de Grenoble, Universität Göttingen, 2010.

[25] A. Rahm and M. Fuchs. The integral homology of PSL_2 of imaginary quadratic integers with non-trivial class group. *Preprint*, arXiv:0903.4517, 2009.

[26] R. Riley. Applications of a computer implementation of Poincaré's theorem on fundamental polyhedra. *Mathematics of Computation*, 40(162):607–632, 1983. ISSN 0025-5718.

[27] P. Samuel and A. Silberger. *Algebraic theory of numbers*. Hermann, 1970.

[28] J. Schwermer and K. Vogtman. The Integral Homology of SL_2 and PSL_2 of Euclidean Quadratic Imaginary Integers. *Comment. Math. Helvetici*, 58:573–598, 1983.

[29] G. Shimura. *Introduction to the Arithmetic Theory of Automorphic Functions*. Princeton Univ Pr, 1971.

[30] W. A. Stein. *Modular forms, a computational approach*. American Mathematical Society, 2007.

[31] W. A. Stein. *Sage: Open Source Mathematical Software (Version 4.6)*. The Sage Group, 2010. `http://www.sagemath.org`.

[32] R. Swan. Generators and Relations for certain Special Linear Groups. *Advances in Math*, 6:1–77, 1971.

[33] *PARI/GP, version* `2.4.3`. The PARI Group, Bordeaux, 2010. Available from `http://pari.math.u-bordeaux.fr/`.

[34] K. Vogtmann. Rational homology of Bianchi groups. *Mathematische Annalen*, 272(3):399–419, 1985. ISSN 0025-5831.

[35] E. Whitley. *Modular forms and elliptic curves over imaginary quadratic fields*. PhD thesis, University of Exeter, 1990.

[36] D. Yasaki. Hyperbolic tessellations associated to Bianchi groups. *LNCS 6197, ANTS-IX Proceedings*, pages 385–396, 2010.